COGNOS®

Cognos Impromptu(R)

Administrator's Guide

Product Information

This document applies to Impromptu (R) Version 7.1 and may also apply to subsequent releases. To check for newer versions of this document, visit the Cognos support Web site (http://support.cognos.com).

Copyright

Table of Contents

Welcome

What Is In This Document

The Impromptu *Administrator's Guide* shows you how to design, create, and maintain an Impromptu environment that meets the needs of your users.

Other Information

Our documentation includes user guides, tutorial guides, reference books, and other pieces to meet the needs of our varied audience.

All information is available in online help. Online help is available from the Help menu and Help button in Windows products.

The information in each online help system is available in online book format (PDF). However, the information from a given help system may be divided into more than one online book. Use online books when you want a printed version of a document or when you want to search the whole document. You can print selected pages, a section, or the whole book. Cognos grants you a non-exclusive, non-transferable license to use, copy, and reproduce the copyright materials, in printed or electronic format, solely for the purpose of providing internal training on, operating, and maintaining the Cognos software.

In Windows products, online books are available from the Windows Start menu (Cognos) and from the product Help menu (Books for Printing). All online books are available on the Cognos documentation CD. You can also read the product readme files and the installation guides directly from the Cognos product CDs.

Only the installation guides are available as printed documents.

An annotated list of other documentation, the *Documentation Roadmap*, is available from the Windows Start menu or the Impromptu Help menu.

Questions or Comments?

For the fastest response to questions about using Impromptu, contact customer support.

For additional technical information about using Impromptu, visit the Cognos support Web site (http://support.cognos.com).

The software_environments.html file accessible from the /support/products part of this Web site lists the operating systems, browsers, Web servers, directory servers, database servers, and OLAP servers currently supported by Cognos products. For information about supported datatypes, see *Deploying Impromptu Applications*.

Part 1: Getting Started

This part of the guide includes

- an overview of your role as an Impromptu administrator
- a tutorial to give you hands-on experience performing basic administrator tasks

Chapter 1: An Administrator’s Overview

This chapter describes your role as an Impromptu administrator. It explains the importance of

- analyzing user requirements
- creating catalogs
- making joins
- organizing folders
- setting up user classes and security
- creating calculations, conditions, and prompts
- automating tasks
- optimizing performance
- integrating with Cognos BI

The Administrator’s Role

As an administrator, you create and maintain an Impromptu environment that is meaningful to your users, making it as easy as possible for them to produce the reports they need.

To do this, you must be involved in all aspects of Impromptu, from connecting to the corporate database; to setting up effective catalogs, joins, folders, and filters; to ensuring that adequate security procedures are in place. You will also want to make sure that Impromptu is set up in a way that maximizes its performance and makes your job easier.

The Keys to Effective Administration

Before starting to administer Impromptu, you should know what your most important tasks are, and what they involve. They include

- analyzing user requirements
- creating one or more catalogs
- making joins between database tables
- organizing folders into a "business view" that makes sense to your users
- setting up appropriate user classes and security
- creating commonly-used calculations, conditions, and prompts
- automating tasks to maximize efficiency
- optimizing Impromptu’s performance

If you use Impromptu as a component of Cognos BI, you need to know how to integrate Impromptu with other Cognos products.

Analyzing User Requirements

Before designing an Impromptu environment, you should know whose business problems you are solving. It is essential that you know who your users are, what types of reports they need, and what data must be included in the reports.

In Chapter 2, "An Impromptu Tutorial," you will learn more about setting up Impromptu to reflect users’ needs. For more information about user requirements, see "Analyzing User Requirements" (p. 29).

Creating Catalogs

As an administrator, you must create at least one catalog containing the information users need to access data from a database. Before creating a catalog, it is important to know the type of catalog to create, and the database tables to include.

Chapter 2, "An Impromptu Tutorial," will give you hands-on experience creating and working with a catalog. For more information about creating catalogs, see "Creating Catalogs" (p. 33).

Making Joins

Joins are links between database tables. They enable you to relate data in one table to data in another. Impromptu has the tools you need to make, edit, analyze, and test joins. Before making joins, you should decide what joins you need. You should also be aware of potential problems with joins, and how to avoid or solve them.

For more information about joins, see "Making Joins" (p. 39).

Organizing Folders

In an Impromptu catalog, tables are presented as folders. Within each folder, Impromptu displays the columns for the table on which the folder is based. You can make it easier for users to find data they need by tailoring your catalog’s folders to match the users’ information needs, presenting them with a "business view" rather than a "database view" of their folders.

Chapter 2, "An Impromptu Tutorial," provides a hands-on introduction to folders that includes renaming, moving, creating, and organizing folders. For more information about folders, see "Organizing Folders" (p. 47).

Setting Up User Classes and Security

An Impromptu catalog can include many different user classes. Typically, you define a different class for each type of user in your organization, based on user requirements. Impromptu provides security features to accommodate different data access privileges for different user groups. You can also use Access Manager, a set of tools that provide security services for all your Cognos products, to control catalog security.

In Chapter 2, "An Impromptu Tutorial," you will create and modify user classes. For more information about user classes and security, see "Setting Up User Classes and Security" (p. 53).

Creating Calculations, Conditions, and Prompts

You can create calculations, conditions, and prompts within a catalog to help standardize reporting results. This saves time and ensures reporting consistency throughout your organization. You control calculations, conditions, and prompts centrally, so all reports are updated automatically as your calculations, conditions, and prompts evolve.

In Chapter 2, "An Impromptu Tutorial," you will create a condition to filter data, define a calculated column, and create two report prompts. For more information about calculations, conditions, and prompts, see "Creating Calculations, Conditions, and Prompts" (p. 63).

Automating Tasks

As an administrator, you should be aware of Impromptu’s time-saving automation features. These include inheritance features that enable you to easily maintain and update distributed reports, CognosScript Editor and CognosScript Dialog Editor for automating complex or repetitive tasks, and Cognos Scheduler for processing during off hours.

For more information about automating and updating Impromptu, see "Automating and Updating Impromptu" (p. 69).

Optimizing Performance

Where and when you want processing to occur will depend on your environment, including your company’s networking capabilities and its types of users. Impromptu provides client/database-server balancing, table weighting, and qualification options to help ensure that you get the best possible performance. You can also change Governor settings that affect processing, and use alternative data sources such as snapshots, thumbnails, and HotFiles.

For more information about optimizing performance, see "Optimizing Performance" (p. 75).

Integrating Impromptu with Other Cognos Products

Impromptu is a component of Cognos BI, a complete business intelligence solution for your enterprise. Cognos BI also includes

- Cognos Query, an ad hoc reporting tool for the Web
- PowerPlay, an OLAP reporting tool for Windows and the Web
- Impromptu Web Reports, a managed reporting environment for the Web
- Cognos Upfront, a secure BI Web portal

Cognos BI uses Access Manager for centralized security control, and Architect to create and maintain a business model that all Cognos products can use.

For more information about integrating Impromptu and other Cognos products, see "Integrating Impromptu with Other Cognos Products" (p. 91).

Where to Next?

Now that you know what your role as an administrator involves, we suggest that you work through Chapter 2, "An Impromptu Tutorial." It won’t take long, and it will provide you with a strong foundation to help you better understand the information in Part 2 of this guide, "Administering Impromptu."

Chapter 2: An Impromptu Tutorial

One of your principal tasks as an administrator is to create and maintain one or more catalogs, so users can access the data they need. In this tutorial, you will

- create a catalog
- organize the catalog's folders to present a business view of your data
- define user classes for the different groups of people who will use the catalog
- filter data for users, so it is easier for them to find what they need
- create a calculated column needed to show a low margin value
- define catalog prompts that can be used in reports
- limit the time users can run queries, to manage the impact of Impromptu processing on your network and database
- test your catalog from the users' perspective, to make sure it meets their needs

What is a Catalog?

A catalog is a file containing the information that Impromptu users need to create reports. Catalogs are central to the operation of Impromptu, and creating a catalog is one of your key tasks as an administrator. A properly-created and maintained catalog results in a reporting environment that makes sense to your users, and makes administering Impromptu easier for you.

For more information about Impromptu catalogs, including a description of each type of catalog you can create, see "Creating Catalogs" (p. 33).

Creating a Catalog

In this tutorial, you are the database administrator for a camping equipment company called The Great Outdoors. You have met with your users—sales managers and salespeople—and know that they need to track sales, inventory levels, and return rates for your products.

For more information about analyzing requirements before creating a catalog, see "Analyzing User Requirements" (p. 29).

Make sure that Impromptu is running on your PC, then use the following steps to create a catalog and attach it to the OUTDOORS database that comes with Impromptu.

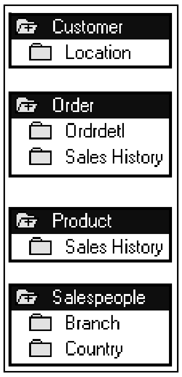
1. From the Catalog menu, click New.
   The New Catalog dialog box appears.
2. In the File Name box, type Sales to name the catalog.
3. In the Description box, type Sales & Returns.
4. In the Catalog Type box, select Shared.
5. In the Database Name box, select OUTDOORS.
6. Select the Include All Tables option to include all the tables and columns from the database in the catalog.
   When setting up your own catalog, you will probably want to select specific tables rather than include them all. If your database is large, and you include unneeded tables in a catalog, it can have a negative effect on processing speed, and make it difficult for users to find the data they need.

7. Click OK.
   The Joins dialog box appears. Table joins are needed to create reports with data from more than one table. You could use the Joins dialog box now, to specify joins for your catalog. For the tutorial, though, you will accept the default, which is no joins.
8. Click OK to close the Joins dialog box.

When you create your own catalog, you will probably want to make joins. For more information about joins, see "Making Joins" (p. 39).

Creating a Business View of the Catalog's Folders

Each table in a catalog is displayed as a folder, with its contents shown as columns. As a database administrator, you understand the organizational structure of your database. But most report users do not, and would be lost trying to create reports from information organized according to a "database view" of your organization.



For example, while the OUTDOORS database contains tables such as ORDER, ORDRDETL, REP, and CUSTSITE, your sales managers are interested in Sales Orders, Customers, and Products.

To make it easier for users to find the data they need, you can reorganize folders into a "business view" that is meaningful to the users. In this tutorial, you will organize the folders in the Sales and Returns catalog as shown in the graphic on the right.

To do this, you will:

- rename the Rep and Custsite folders to Salespeople and Location
- move the Location, Orderdetl, Branch, and Country folders
- create and organize a new folder for Sales History

Renaming the Rep and Custsite Folders

You will change the names of the Rep and Custsite folders to make them more meaningful.

1. From the Catalog menu, click Folders.
   The Folders dialog box appears.
2. Select the Rep folder.
3. Click the Edit button.
   The Edit Name dialog box appears.
4. In the Name box, type Salespeople.
5. Click OK.
   The Rep folder has been renamed Salespeople.
6. Repeat steps 2 to 5 to change the name of the Custsite folder to Location.

Moving the Location, Ordrdetl, Branch, and Country Folders

To make it easier for users to find information when they build reports, you will move some folders inside others.

Here is what you will do:

So users can find information on ...	**You will group these folders ...**
Customers	Customer and Location
Orders	Order and Ordrdetl
Salespeople	Salespeople, Branch, and Country

1. In the Folders dialog box, select the Location folder.
2. Click Cut.
   Impromptu cuts the Location folder from the catalog and places it on the Clipboard.
3. Double-click the Customer folder to open it.
4. Click Paste.
   Impromptu pastes the Location folder into the Customer folder.
5. Double-click the Customer folder to close it.

If you make a mistake in the next two steps, click the Undo button and start over.

6. Repeat steps 1 to 5 to move the Ordrdetl folder into the Order folder.
7. Repeat steps 1 to 5 again, to move the Branch and Country folders into the Salespeople folder.

Leave the Folders dialog box open for the next step in the tutorial.

Creating and Organizing a New Folder

Sales history information is contained in the Product folder. But users often look for it in the Order folder. To ensure they can find the information in either the Product or Order folder, you will:

- create a folder specifically for Sales History
- move the sales history data into the Sales History folder
- copy the Sales History folder into the Order folder

Creating a Folder for Sales History

1. In the Folders dialog box, double-click the Product folder to open it.
2. Click the New Folder button to create a new folder.
   The New Folder dialog box appears.
3. In the Name box, type Sales History.
4. Click OK.

Moving Data into the Sales History Folder

Now that you have created a new folder for sales history, you will move the sales history information into it.

1. In the Folders dialog box, select the Sales 92, Sales 93, Sales 94, and Sales 95 columns.
2. Click Cut.
   Impromptu cuts the selected columns from the catalog and places them on the Clipboard.
3. Double-click the Sales History folder to open it.
4. Click Paste.
   Impromptu pastes the columns into the Sales History folder.

Copying the Sales History Folder into Another Folder

Your sales history information is now in a Sales History folder. But the Sales History folder is still in the Product folder, and you want it in the Order folder as well. You will copy the Sales History folder to the Order folder.

1. In the Folders dialog box, double-click the Sales History folder to close it.
2. With the Sales History folder selected, click Copy.
   Impromptu copies the Sales History folder to the Clipboard.
3. Double-click the Product folder to close it.
4. Double-click the Order folder to open it.
5. Click Paste.
   Impromptu pastes a copy of the Sales History folder into the Order folder.
6. Click OK to close the Folders dialog box.

You have created a Sales and Returns catalog and organized it to match your business view, making it easier for users to find the information they need.

For more information about using folders to create a business view of your catalog, see "Organizing Folders" (p. 47).

Defining User Classes

In Impromptu, each user community is called a user class. If you group user communities into classes, you can create one catalog for many groups; minimizing your workload, controlling the data that each user class is allowed to work with, and ensuring that the catalog meets each unique set of user needs.

Four user groups, each with its own set of requirements and privileges, will use the Sales and Returns catalog you have created:

- North American sales managers
- North American salespeople
- European sales managers
- European salespeople

Creating a User Class for North American Managers

You need two user classes for the North American sales region—one for sales managers, the other for salespeople.

1. From the Catalog menu, click User Profiles.
   The User Profiles dialog box appears.
2. Click the User Classes tab.
   Each user class is based on another user class. You have created a Shared catalog that will be used by more than one user, so you already have two user classes—Creator and User. Only the Creator user class is visible, because the User user class is based on the Creator user class.
3. Double-click Creator to show the User icon.
   You need to define four user classes, all based on Creator. To do this, you will rename the existing User class, then add three more user classes.
4. Click the User user class.
5. Click the Edit button.
   The Edit User Class dialog box appears.
6. In the Class Name box, type North American Managers.
   It isn't necessary for this tutorial, but you could assign a password for each user class by typing it in the User Class Password box.
7. Click OK to close the Edit User Class dialog box.

You should now have two user classes in the User Classes dialog box—Creator, and North American Managers.

Creating a User Class for North American Salespeople

You want to base the user class for salespeople on the one for their managers, so the salespeople will inherit the same privileges and restrictions as the managers.

1. Select the North American Managers user class.
2. Click the Add button.
   The Add User Class dialog box appears.
   The text above the Class Name box tells you that the new user class will be based on the North American Managers class. The user class that you create now will inherit all of the restrictions and privileges of the North American Managers class.
3. In the Class Name box, type North American Salespeople.
4. Click OK.

You have created two user classes for the North American sales region—North American Managers, and North American Salespeople.

Creating User Classes for Europe

You need to create two more user classes for the European sales region. Using the skills you learned adding the North American user classes:

- create a user class called European Managers based on the Creator user class
- create a user class called European Salespeople based on the European Managers user class

When you have created these classes, leave the User Profiles dialog box open for the next part of the tutorial.

For more information about creating and modifying user classes in Impromptu, see "Setting Up User Classes and Security" (p. 53).

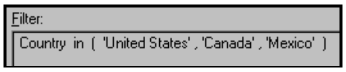
Filtering Data

Your North American users need to see data for only North America, while your European users need data for only Europe. Data for other regions is irrelevant, and users do not want to have to filter the data in their reports manually.

You can easily create filters for user classes in the catalog, so that users see only the data you want them to work with. You can also use filters to set up security by value in Impromptu. In this part of the tutorial, you will create a filter that is attached to the Country column in the Country table. Whenever your users access this column, the filter will be applied automatically to their data.

In Impromptu, filters are also referred to as "filtering conditions," or simply "conditions."

The filter you create for the North American Managers user class will look like this:



Country in ('United States' , 'Canada' , 'Mexico')

You will create this filter for the sales managers only, since the salespeople will inherit it automatically.

1. In the User Profiles dialog box, click the Filters tab.
2. In the User Class box, select North American Managers.
3. In the Catalog Tables box, select Country.
4. Click the Edit button.
   The User Profile Filter dialog box appears.

5. In the Available Components box, double-click Catalog Columns.
   The Catalog dialog box appears.
   You want your users to see only data only for their regions, so you will attach the filter to the Country column.
6. Double-click the Salespeople folder, double-click the Country folder, then double-click the Country column.
   The word Country appears in the Expression window, with the cursor located on the ellipsis (...) following the word.

in

7. In the Available Components box, double-click the "in" option to start entering your filter.

(

8. Double-click the "open bracket" option to insert an open bracket.

string

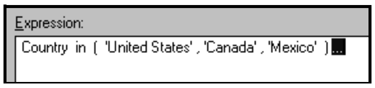
9. Double-click the "string" option to insert a string in the Expression box.
   You will use the string to specify a country for the filter.
10. Type United States between the quotation marks.
11. In the Expression box, click the ellipsis (...) to highlight it.

,

12. Double-click the "comma" option to insert a comma.
13. Repeat steps 9 to 12 to insert a string for Canada.
14. Repeat steps 9 to 11 to insert a string for Mexico.

)

15. Double-click the "close bracket" option to complete the filter.
    Your filter should look like this:



Expression:

Country in ('United States' , 'Canada' , 'Mexico')

16. Click OK to close the User Profile Filter dialog box.

Now, when users in the North American Managers user class access the Country column in a report, only data for North America is displayed. Remember, this condition applies automatically to the North American Salespeople user class too, because this user class inherits properties from the North American Managers user class.

If you want more practice creating filters, use the skills you learned creating a filter for the North American Managers user class to create a filter for the European Managers user class. Create the filter on the Country column so that these users will see only data for France, Germany, Spain, Sweden, and the United Kingdom. When you have finished, leave the User Profiles dialog box open for the next part of the tutorial.

For more information about filters see "Creating Calculations, Conditions, and Prompts" (p. 63).

Sharing Your Administrative Tasks

You want to share some of your administrative tasks with the sales managers, but not with the salespeople. You will allow the managers to organize catalog contents and create user classes for their divisions, but you will not provide these capabilities to the salespeople.

1. In the User Profiles dialog box, click the Governor tab.
2. In the User Class box, select North American Managers.
3. If it is not already selected, select the Edit Folders check box to enable this user class to organize the catalog contents.
4. If it is not already selected, select the Add/Modify User Classes check box to enable this user class to create other user classes.
   Users in the North American Managers user class are now able to create user classes based on their own user class.
5. Repeat steps 2 to 4 for the European Managers user class.
   The two user classes for sales managers can now share some of your administrative tasks.
   You don't want the salespeople sharing these responsibilities, so you need to prevent them from organizing the catalog contents and creating user classes.

6. From the User Classes box, select North American Salespeople.
7. If it is not already cleared, clear the Edit Folders check box to prevent this user class from organizing the catalog contents.
8. If it is not already cleared, clear the Add/Modify User Classes check box to prevent this user class from creating other user classes.
9. Repeat steps 6 to 8 for the European Salespeople user class. You will continue to make changes after you have done this, so DO NOT click OK.

You have now prevented users in some user classes from performing administrative tasks, while enabling users in other user classes to perform these tasks.

Limiting the Query Execution Time

On a network that supports a large number of users, performance can become an issue if users frequently retrieve large amounts of data. As the administrator, you can avoid this by setting some operational constraints by user class. For example, you can use Governor settings to limit the time that users can spend running queries.

1. In the User Class box, select North American Managers.
2. In the Query Execution Time Limits box, type 15 in the Warn At: Minutes box.
3. In the Stop At: Minutes box, type 30.

   When users run a query using this catalog, they will be warned when the query has run for 15 minutes, and the query will be stopped when it has run for 30 minutes.
4. Repeat steps 1 to 3 for the European Managers user class.
5. Click OK to save your changes and close the User Profiles dialog box.

Remember, the North American Salespeople and European Salespeople user classes inherit this restriction from the manager classes.

Impromptu also provides other ways to control execution time, including the UNIX Request Server, Hotfiles, and snapshots. The database you use might have features for controlling execution time as well.

For more information about optimizing Impromptu’s performance, see "Alternative Data Sources" (p. 84).

Creating a Calculated Column

A calculation is an expression that calculates a new data item from existing data items. Creating a calculation and storing it in a catalog saves you and your users time, because you don’t have to re-create frequently used calculations. As well, you maintain control over the definition of the calculation, since you create it and others use it. If you change the calculation, all reports using the calculation are updated automatically.

Your users need a calculation that shows low margin, so they can track products with a profit margin that is too low.

1. From the Catalog menu, click Folders.

   The Folders dialog box appears.
2. Click the New Calculation button.

   The New Calculation dialog box appears.
3. In the Name box, type Low Margin.

   The options in the Available Components box work the same as those in the User Profile Filter dialog box. If you need help in the next step, see "Filtering Data" (p. 21).

4. Using the options in the Available Components box, and the tips that follow the calculation, build this calculation:

   ```
   if ((Price - Prod Cost) <= 0.20 * Price) then ('Low Margin') else NULL
   ```

   Tips for building the calculation:

   - The first left parenthesis is for the entire IF expression; the second is for the first calculation: Price - Prod Cost.
   - Price is a column in the Ordrdetl folder, which is in the Order folder.
   - Prod Cost is a column in the Product folder.
   - Add 0.20 by double-clicking Number in the Available Components box and then typing the number. When you have done this, click the ellipsis (...) following the number.
   - Add "Low Margin" by double-clicking Value in the Available Components box, then double-clicking String, then typing the words Low Margin between the quotation marks. When you have done this, click the ellipsis (...) following the string.
5. Click OK to close the New Calculation dialog box.

   The Low Margin calculated column appears in the Folders dialog box. Leave the Folders dialog box open for the next step.

You have created a calculated column for all your users.

Creating Catalog Prompts

When users define reports, they can include prompts to specify data selection criteria. To help ensure consistency, and to retain centralized control over commonly used prompts, you can store prompts in your catalog. To make it easy for users to find the prompts they need, you can store them in their own folder, or in folders containing related business information.

You will define two commonly used prompts—Start Date and End Date—to be used when prompting for a date range during report generation.

1. In the Folders dialog box, click the New Prompt button.

   The Prompt Definition dialog box appears.
2. For the Name, type Start Date.
3. In the Type list, select Type In.
4. For the Message, type Enter the Start Date of the Date Range.
5. In the Data Type list, select date.
6. For the Default Value, type 1996-01-01.
7. Click OK.

   The Start Date prompt appears in the Folders window.
8. Repeat steps 1 through 7 to define an End Date prompt. For the Name, type End Date. For the Message, type Enter the End Date of the Date Range. For the Default Value, type 1998-01-01.
9. Click OK to close the Folders dialog box.

The prompts you have defined are available to all your users, but users cannot delete or modify the prompts.

For more information about calculations and prompts, see "Creating Calculations, Conditions, and Prompts" (p. 63).

Testing Your Catalog

To test how the Sales and Returns catalog works for your users, you will change your user class to one of the user classes you created.

1. From the Catalog menu, choose User Class.

   The Catalog Logon dialog box appears.
2. In the User Class box, select North American Managers or European Managers.

3. Click OK to close the User Class dialog box.
4. Click the Catalog menu.
   Notice that commands that are unavailable to the North American and European Managers are grayed. For example, they cannot update the catalog or access the Tables or Joins selections. But they can set up user classes based on their own class, and they can organize Folders.
5. From the Catalog menu, click User Profiles.
   The User Profiles dialog box appears.
6. Click the Governor tab.
   As you can see, the only user class the North American Managers can view and change is North American Salespeople, and the only class the European Managers can view and change is European Salespeople. The sales managers cannot change their own access privileges.
7. Click OK.
8. Return to the Catalog menu and change the user class to North American Salespeople or European Salespeople.
9. Look at the Catalog menu again. Notice that the Salespeople user class is more restricted than the Manager user class. Unlike managers, the salespeople cannot modify Folders or User Profiles.

You can test the catalog for each of your user classes, checking to make sure that it meets users' requirements before giving it to them. You can then refine the catalog before users work with it. You can also easily change the catalog after giving it to your users. Your users will get the latest version of the catalog automatically.

A Quick Review

If you worked through the tutorial from beginning to end, then you

- created an Impromptu catalog
- renamed and moved some of its folders
- created and organized a new folder
- defined user classes
- created filters on user classes
- established processing time limits
- created a calculated column
- created catalog prompts
- tested your catalog from a user's perspective

In the process, you have become familiar with at least part of the Impromptu user interface. You also have a good foundation to help you make the most of the information in Part 2 of this guide, "Administering Impromptu."

Getting Online Help

For more information about catalogs, including step-by-step instructions for performing tasks associated with catalogs, see the Impromptu online help.

Part 2: Administering Impromptu

This part of the guide provides information to help you administer Impromptu effectively. It includes chapters on

- analyzing user requirements
- creating catalogs
- making joins
- organizing folders
- setting up user classes and security
- creating calculations, conditions, and prompts
- automating and updating Impromptu
- optimizing performance
- integrating Impromptu with other Cognos products

Chapter 3: Analyzing User Requirements

Administering Impromptu effectively means ensuring that it meets the needs of its users. Before developing your Impromptu environment, you should know who your users are, and what their requirements are. This chapter will help you do that.

Who Are Your Users?

Before you create an Impromptu environment, evaluate the needs of the people in your organization who will use Impromptu.

Users are typically knowledge workers who need to generate a variety of reports. Depending on the privileges you assign them, and the type of environment you set up for them, they might or might not be able to

- print reports
- create their own reports
- customize their environment by modifying a catalog
- work with automated tasks or automate their own tasks
- provide limited administrative support to other users
- create reports accessible to disabled users

To determine user requirements, you might want to interview representatives of different user groups to assess their levels of knowledge and technical expertise, and to determine their functional areas and hierarchical levels. You might also consider developing a questionnaire designed to determine each group's needs.

Determining User Classes

In Impromptu, each user community makes up a user class. When planning your Impromptu environment, begin by determining how many classes you have, and what their particular needs are.

Typically, employees other than those in the Information Systems department are not able to explain their needs in terms of the structure of your database. As an administrator, it is up to you to translate their requirements to the database level.

Different types of users typically have different reporting requirements and different privileges or restrictions when it comes to accessing data and modifying their Impromptu environment. For example, a sales department might have sales managers and salespeople with very different reporting requirements and data access privileges.

If you completed Chapter 2, "An Impromptu Tutorial," then you already have some experience setting up environments for different users. If you didn't complete the tutorial, consider doing it after reading this chapter.

Determining the Domain of Data

Once you know the user classes you need, you can determine the domain of data in your database to which each user class needs access. Knowing this will help you determine the type of Impromptu catalog to set up, the tables to include in the catalog, the joins to make between tables, and the folders and columns to include.

When you have discussed the users' requirements with them, and analyzed their needs, you can systematically break the requirements down into "What information does the user want, by what." What they want to see translates into data columns in Impromptu. The "WHAT they want BY" determines the column groupings they require.

Example: A Sales Analysis

The domain of data for a sales group might look like this:

1 Products	**2 Time**	**3 Organizations**	**4 Customers**	**5 Indicators**
Product lines	Years	Sales Divisions	Sales Rank	Units Ordered
Brands	Quarters	Sales Districts	Customers	Number of Orders
Products	Months	Sales Reps		Order Value $
Stock Keeping Unit (SKU)				Revenue
				Discount
				Average Selling Price

The columns in the table represent the information users will need:

1. Product line, brand, product, and stock keeping unit information.
2. Sales dates, for analysis of time in years, quarters, and months.
3. Sales divisions, districts, and representatives.
4. Sales rankings by customer.
5. What they are measuring—units ordered, number of orders, booked value of orders, revenue, discount, and average selling price.

An additional report might be needed with customer details such as name, phone number, balance outstanding, and similar information.

Determining the Data Elements

When you have identified the domain of data and the reports needed, you can determine what data elements are involved, and where they are in the database. You will need to determine how to map the elements defined in the domain of data to elements defined in the database, keeping in mind that:

- some elements may not be available
- some elements need to be derived
- some elements can be in more than one place, or have different names

When doing this type of analysis, you might find it helpful to make a sketch of your database, showing the table joins you need to ensure that users can access the data they need.

Planning on Change

It is important to be as precise as possible when determining user requirements, but it is next to impossible to define them exactly. Requirements typically are not fixed; they change from day to day with changing business issues. As a result, it would be a mistake to try to definitively document each report or view that your users need—it would take too long, and you would never be finished.

Instead, when planning your Impromptu environment, keep in mind that you are documenting the domain of data that your users think they need, plus some representative standard reports based on this domain. Once these reports have been built, users can verify whether the reports meet their needs.

Analyzing user needs is an iterative process that continues throughout the life of your organization. As users become proficient with Impromptu, they will probably create their own reports, and take on more responsibilities. But you will likely still need to add or remove data items from the users' domain of data.

For more information about user classes see "Setting Up User Classes and Security" (p. 53).

Chapter 4: Creating Catalogs

A well-designed catalog is essential to the successful operation of Impromptu, and can be the key to its ease of use. This chapter tells you

- what a catalog is
- what its benefits are
- what a catalog contains
- how many catalogs you should create
- what type(s) of catalog(s) to create
- what effect a catalog's size has
- what you should think about before creating a catalog
- how to start creating a catalog

It also includes catalog tips and guidelines, and tells you where to find more information if you need it.

What is a Catalog?

A catalog is an Impromptu file (*.cat) that you, as an administrator, must create. A catalog is important because it contains the information users need to access data in a meaningful and controlled way. Essentially, it is a bridge between the source data and the Impromptu user.

A good way to learn about catalogs is to create one and explore the options in the catalog's dialog boxes. If you haven't completed Chapter 2, "An Impromptu Tutorial," consider doing it now. You might also want to experiment with The Great Outdoors catalog (Great Outdoors Sales Data.cat) that comes with Impromptu.

What Are a Catalog's Benefits?

A catalog has benefits for both you and the user. As an administrator:

- it means you can serve different user communities without having to modify your database
- it allows you to specify what information is available to users and how it is accessed
- it can provide you with a single point of maintenance for your entire Impromptu environment

From the users' perspective, a catalog:

- insulates them from the database, making database training unnecessary
- presents them with an intuitive, business view of their data

What Does a Catalog Contain?

A catalog contains:

- Folders—meaningful groups of information representing columns from one or more tables
- Columns—individual data elements that can appear in one or more folders
- Calculations—expressions used to compute required values from existing data
- Conditions—used to filter information so that only a certain type of information is displayed
- Prompts—pre-defined selection criteria prompts that users can include in reports they create
- Other components, such as metadata, a logical database name, join information, and user classes

How Many Catalogs Should You Create?

Impromptu can handle many catalogs. If you have an extremely large amount of data and many tables in your database, then defining more than one catalog might make sense. In this case, a single catalog could be difficult to use. It might take too long to load and navigate the catalog, and join maintenance might become involved.

More catalogs mean more administration and maintenance, though, and as a rule it is advisable to have only one catalog per application per physical location. Impromptu's user classes and security features make it easy to manage a wide variety of user requirements with a single catalog.

For more information about user classes and security, see "Setting Up User Classes and Security" (p. 53).

What Type of Catalog Should You Create?

You can create four different types of Impromptu catalogs:

- personal (the default)
- shared
- distributed (the most common)
- secured

The type to create depends on your environment, the knowledge of your users, and your security requirements. Once you have built a catalog of one type, you can easily change it to another type.

The Personal Catalog

A personal catalog can be used by only one person, and cannot be opened with the User version of Impromptu.

If you are creating a catalog for your own use only, then you will want to create a personal catalog. You might also want to begin the catalog creation process by creating a personal catalog. Then you can test the catalog to ensure that it is set up correctly before converting it to another type of catalog that can be used by others.

The Shared Catalog

A shared catalog can have many users. The users can create their own reports, but cannot change the catalog. With a shared catalog, all your users work with the same catalog.

A shared catalog is useful when:

- you want to avoid problems that can arise when users "personalize" their catalogs, by creating different reports and using different data so that consistency throughout your organization is lost
- you want to store the catalog on a shared drive on a Local Area Network (LAN)
- it is necessary to protect the catalog against modifications, except by you or someone else in the Creator user class
- you want users to be able to create and edit their own reports

If you create a shared catalog, you will want to define different user classes. You will also want to clear the Edit Folders checkbox in the Governor tab of the User Profiles dialog box for users who should not be able to make modifications. By default, users of a shared catalog cannot add to the catalog, but you can assign them this ability using Governor settings.

The disadvantages of a shared catalog are:

- users must have access to the LAN to access the catalog
- you can only maintain the catalog when no-one is using it

The Distributed Catalog

When you create a distributed catalog, you keep a master distributed catalog, and a personal distributed catalog is made from the master for each user. Users can create reports and make changes to their copies of the catalog within the limits of the user classes assigned to them. When you change the master distributed catalog, the changes are automatically applied to each user's personal distributed catalog.

A distributed catalog is useful if:

- users must be able to create and edit reports and append to the catalog contents, but also need the benefit of a common catalog for their group
- users need to be able to work off-line (although they still need access to the database)
- you want to make it easier for users to make changes than it would be with a shared catalog, since a shared catalog requires all users to work from the same catalog
- as an administrator, you want to maintain the catalog centrally

Some disadvantages of a distributed catalog are:

- network traffic is incurred when copying catalogs to user machines
- there may be a short delay when a personal distributed catalog is opened, while it synchronizes with the master distributed catalog

The Secured Catalog

A secured catalog can also be a distributed catalog. But with a secured catalog users have a read-only version of the catalog. They can't change the contents of the catalog or the reports created with the catalog, and they can't create new reports. But they can run reports you have created for them, convert reports to different file formats, and print reports.

With a secured catalog, you maintain the advantage of having centralized control over the catalog, and complete control over reporting against the database.

The disadvantages of a secured catalog are the same as for a shared catalog.

Impromptu Catalog Types: A Summary

This table summarizes the types of Impromptu catalogs you can create.

Type	Description
Personal (The Default)	Can be used by only one person, and cannot be opened with the User version of Impromptu. Create it for your own use, or as the starting point for another type of catalog.
Shared	Can be used by many users, who can create their own reports but cannot change or add to the catalog. You have centralized control, but users must have access to the LAN, and you must take responsibility for making changes to the catalog.
Distributed	Can have many users, each with a read-only version of the catalog. You maintain centralized control of the catalog, but users can alter their own copies within the limits you have set for them.
Secured	Can have many users, each with a read-only version of the catalog. They can run reports you have created, but cannot create new reports or edit existing ones.

What Effect Does a Catalog’s Size Have?

As you include more tables, folders, and security restrictions in a catalog, it becomes larger. The size of a catalog can determine the type of catalog you choose to create.

With a distributed catalog, the larger the master distributed catalog the longer it takes to copy the personal distributed catalog to the user machine, the more traffic there is on the network, and the longer it takes to synchronize the master and personal catalogs.

There are no internal limits to the size of a catalog; that is, it can contain any number of tables. But large catalogs—anything approaching 1 MB in size—can be inefficient, especially on slower machines.

As a catalog increases in size:

- maintenance time increases
- it takes longer to load the catalog
- organizing folders for easy use by users becomes more difficult
- user class security becomes more involved
- join definition becomes more complex, with more chance for join loops
- distributed catalog synchronization takes longer

It is difficult to provide an ideal size for a catalog, or a maximum number of tables in a catalog. It is best to add incrementally and check performance at each step, testing it on a typical end user machine for a real world feel.

Things to Consider Before Creating a Catalog

Before starting to create a catalog, you should be able to answer these questions:

1. What information do you want to include in the catalog?

   For example, do you want to see data for a particular division of your company? If you are interested in creating reports that present only marketing information, then you probably need to access only the tables in the database that contain marketing data.

2. Do you want to create a catalog that contains all the tables from the database, or only selected tables?

   Including all the tables is easier, because you do not have to specify the individual tables to include. But if you include tables that aren’t needed, processing efficiency might decrease, and it might be more difficult for users to find information they need.

   As a rule, include only those tables that must be accessible to users. Some database tables are used strictly for linking other tables. They should be included in the catalog for join purposes, but excluded from the folders. The range of tables to include in any one catalog should always be based on user and reporting requirements.

3. What table joins do you want to set up?

   When you create a catalog, you can join tables so users can retrieve data from more than one table at a time. Although you can edit joins at any time, before starting to create a catalog you should have a join strategy in mind for the catalog.

For more information about making joins, see "Making Joins" (p. 39).

How to Create a Catalog

When you are ready to begin creating a catalog, select New from the Catalog menu to display the New Catalog dialog box, then make the entries needed to create the catalog you want. You can find any help you need online.

Opening a Catalog Created in an Earlier Version of Impromptu

You can open a catalog created in an earlier version of Impromptu with a more recent version of Impromptu, but not the other way around.

When you start to open a catalog created in an earlier version of Impromptu, the Catalog Update dialog box appears.

You can open the catalog as read-only, or upgrade the catalog. If you open a catalog as read-only, you will not be able to modify the catalog. If you upgrade a catalog, you will no longer be able to open the catalog with the version of Impromptu that was used to create the catalog.

During a transition from one version of Impromptu to another, back up all catalogs regularly.

Some Catalog Tips

Here are some tips to keep in mind when creating and maintaining catalogs:

- You can create a Content Overview Report that contains information about the contents of a catalog you have created. It is saved as an ASCII text file, so you can open it with any word processor. It can be a handy tool to help you review what you have done when setting up a catalog. For more information, see the Impromptu online Help.
- Back up your catalogs and reports regularly, even while developing them.
- The description of a master distributed catalog is migrated to the personal distributed catalog of each user. If you change the description of the master, it will only apply to users who open the master version after the change to the description.
- Advise users to save their personal distributed catalogs with the same name as the master. This will minimize confusion that can arise when users have many catalogs in their directory and need to determine which master catalog is being referenced.
- The more folders a catalog has, the larger it is. Avoid repeating folders as subfolders in many different folders. Each copy of the folder increases the size of the catalog. This must be balanced with the need to make the catalog valuable to the user and the need to ease the administrative burden.
- Denying access to a lot of folders for a large number of user classes increases the size of the catalog. When a user has access to a folder, no overhead is added, but denying access increases catalog size. You should monitor this feature closely.

For More About Catalogs

For more information about catalogs, including step-by-step instructions for performing tasks associated with catalogs, see the Impromptu online Help.

Chapter 5: Making Joins

Proper joins ensure that users can access the information they need, and that they do not receive unexpected, incomplete, or incorrect results. This chapter tells you

- what a join is
- how to make and edit joins
- what types of Impromptu joins you can make
- how to analyze joins
- how to solve problems with joins
- how to test joins

It also includes some join strategy tips, and tells you where to find more information about joins if you need it.

What Are Joins?

Joins define the relational links between tables in the physical database. They enable you to relate data in one table to data in another table in the same database, so users can retrieve data from more than one table at a time. This helps ensure that users can access the information they need no matter where it is in the database.

Proper joins are crucial to the success of ad hoc reporting, and are also required for standard reports. Once you define joins, they are available for all users, and do not have to be redefined unless the database structure changes. Users are shielded from having to know anything about how tables relate to one another.

Making and Editing Joins

When you create a catalog, the Joins dialog box appears.

If you planned a join strategy for the catalog you are creating, you can use the Edit tab in the Joins dialog box to make the joins you want. Impromptu's online help includes step-by-step instructions for making joins.

You can also choose to return to the Joins dialog box later, to make or modify joins. Joins are not made automatically, so if you don't make them your catalog will not have any. You can also choose to generate them automatically using the Create Joins box in the Tables dialog box.

Types of Impromptu Joins

You can make six types of Impromptu joins:

- Equi
- Non-Equi
- Outer
- Self (Reflexive)
- Compound
- Complex

Equi Joins

The most common type of join is an equi join. Equi joins retrieve all of the rows from one table and the rows in another table when the values in the linked columns are equal.

For example, the linked columns of a database's Products and Orders tables are the two columns for Product Number. An equi join between these tables would match up equal values in the two Product Number columns, then retrieve the rows from both tables that contain these values. All of the possible combinations of these rows could then be reported on.

Non-Equi Joins

Non-equi joins retrieve all of the rows from one table that meet certain criteria in another table. The criteria are based on expressions that you define using these operators:

- <> not equal to
- < less than
- > greater than
- <= less than or equal to
- >= greater than or equal to

To create a Sales Representative report showing which representatives sold more or less than a given representative, you could compare the values in the sales amount column with a non-equi join. (This would also require a self-join, which is described later in this chapter.)

Outer Joins

Outer joins retrieve rows from one table even if there are no matching rows in another table. They can be used in conjunction with self, non-equi, complex, or compound joins. For example, to create a report showing all sales by sales representative, and include representatives who made no sales at all, you can use an outer join.

An outer join can be a left, right, or full outer join. The side of the outer join determines which rows are reported on, even if there is no match. To determine whether to create a left or right outer join, you must know the relationship between the two tables you are joining. The outer join goes on the master side (which returns all rows), not the detail side. A full outer join retrieves all the rows from both tables, as if you combined a left outer join with a right outer join.

Impromptu provides support for left, right, and full outer joins. In doing so, Impromptu might be required to assist a database which provides less complete support of the SQL-92 definition of outer join processing. This can affect performance. When defining outer joins, please refer to your vendor documentation.

Self Joins (Reflexive Joins)

Self joins are required when you want to join a table to itself, so you can compare data in a column to other data in the same column. To create a self join, you create an alias table, then join it to its source table, typically with a non-equi join.

What is an Alias Table?

An alias table is a catalog table that you create from an existing (source) database table, using a button in the Tables dialog box. The result is two tables with the same contents but different names.

Self Joins: An Example

You might create a self join if you want to create a report displaying all of the employees that report to each manager. Since a manager is also an employee, both the employee and manager data are in the same table.

To make the report possible, you would create an alias table and join it to its source table. To make it easy for users to understand the information they are selecting, you would ensure that the information is properly identified by the alias name. For example, you could name an alias of "Employees" as "Managers."

Compound Joins

Compound joins are useful where multiple columns generate a unique key to join to another table. Sometimes a relational database has a unique key, which is a combination of two or more columns. A compound join can also be used in combination with a non-equi, self, or outer join.

Complex Joins

Complex joins use an expression created with the Expression Editor to join tables. For example:

```
(Sales.prod_code=Substring(Products.prod_code, 1, 4)
```

They are not used frequently, although they can be used in conjunction with a non-equi or outer join. Use a complex join, for example, when your organization acquires another organization, and you must consolidate information for the two organizations.

When defining complex joins, be aware of the interaction between Impromptu and your database. If you use internal expressions or functions (those with the icon indicating they could be used in Impromptu), processing could end up being done by Impromptu rather than the database. This could have a negative impact on performance.

Impromptu Join Types: A *

This table summarizes the types of Impromptu joins you can create.

Type	Description
Equi	Retrieves all of the rows from one table and the rows in another table when the values of the linked columns are equal. The most common join.
Non-Equi	Retrieves all of the rows from one table that meet certain criteria in another table. The criteria are defined using expressions.
Outer	Retrieves rows from one table even if there are no matching rows in another table.
Self (Reflexive)	Required to compare data in a table column to itself. Can be used only by defining an alias table.
Compound	Useful where two columns will generate a unique key to join to another table. Can also be used in combination with a non-equi, self, or outer join.
Complex	Uses an expression created with the Expression Editor to join tables. Not used frequently.

Analyzing Joins

Once you have created a join structure, you can use the Analyze tab in the Joins dialog box to find and fix problems that would result in unexpected reporting results.

If there is a problem in a join structure, a message in the Analyze tab describes it. You can choose to solve the problem using one of:

- Create Alias
- Edit Join
- Remove Join

You can also click Next to skip the problem, or Restart to return to the first problem. When you solve a problem, the next one appears automatically. When you have solved all of the problems, the Exception area of the Analyze tab displays the message "No exceptions were found."

The Analyze feature looks for unjoined (orphan) tables and for loops in the join structure. When it finds a problem, it displays a message such as:

- No Spanning Tree
- Isolated tables
- Loop

No Spanning Tree/Isolated Table

If the message displayed in the Analyze tab is

```
Exception: NO SPANNING TREE
There is no single spanning tree for all tables
```

or

```
Exception: ISOLATED TABLE
There are no paths defined for this table
```

then there are single tables, or groups of joined tables, that are not linked to the other tables in the join structure.

In the case of No Spanning Tree, a join is missing between groups of joined tables. In the case of an Isolated Table, an orphan table exists. Before you can include information from the unjoined groups or orphaned tables in reports, you must complete the join structure.

Loop

If the message displayed indicates that there is a loop in a join structure, then two or more tables are linked together in multiple ways, providing more than one pathway for accessing tables in the database. The culprit in the case of a loop join is often a lookup table, because it is joined to many tables in the catalog.

Loop joins can cause the joins generated to be less than optimal, decreasing performance. They might also cause you to receive unexpected results.

Solving Join Problems

To solve No Single Spanning Tree and Isolated Table join problems, you can:

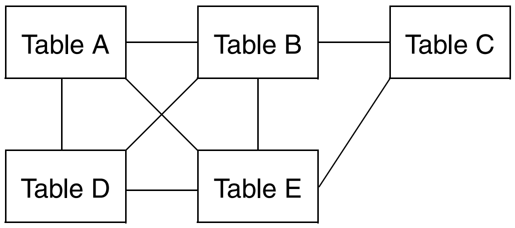
- Join the isolated table(s) with the rest of the join structure. This is usually recommended.
- Leave the join structure as is. It might not be necessary to join the isolated table, because some reports only require information from a single table.

To solve loop join problems, you can:

- Leave the join structure as is. Impromptu automatically tries to use the shortest route in joining multiple tables. Leaving the loop structure unchanged may be an acceptable option if the shortest route is the most efficient. But the structure of the database might work against this option, depending on the type of data and the use of indexes.
- Remove a table from the catalog. This solution requires a re-evaluation of the amount of data that will be included in the catalog and the reporting capabilities possible with the catalog.
- Remove one or more joins (see the example below). Review your data first, to determine which join is unnecessary. Removing the wrong join might result in duplication of data in a query, and inefficient data retrieval.
- Create alias tables with separate joins defined for each alias table (see the example below). This strategy requires additional maintenance in both defining the joins and maintaining the folders. In addition, it may result in additional folders which can add to user confusion.

Solving a Loop Join: An Example

The figure below shows an example of a loop join in which tables A and C can be joined through either table B or table E.

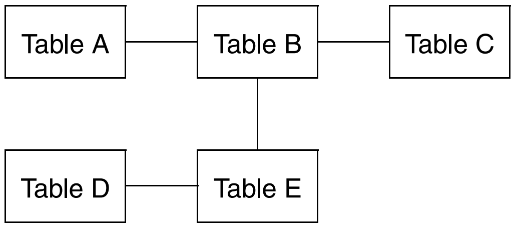


The result returned when going through table B might be significantly different from the one returned when going through table E. While both queries are valid, only one will return the result that you want.

A query might be using the desired path until one additional field is added from a loookup table referenced by more than one of the tables in the original query. The query then contains a loop join, and may use the non-desired path. In this case, the join Analyzer would show a message telling you there is a loop join.

Resolving the loop by editing the join stategy

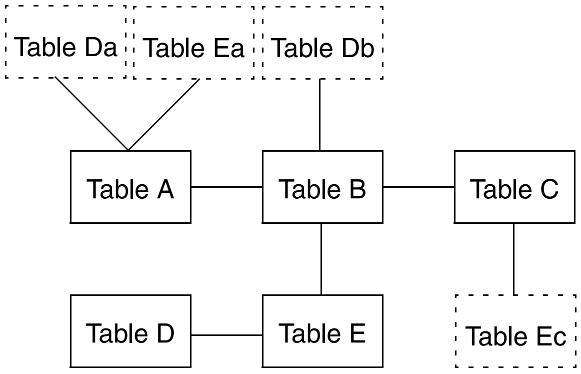
The figure below shows how the loop join was resolved by removing unnecessary joins.



Now, tables A and C can only be joined through table B, and the desired path will be used.

Resolving the loop with alias tables

The figure below shows how the loop join was resolved using alias tables, represented in the figure with dotted borders.



We have aliased the lookup tables D and E in order to have one instance for each additional table to which the look-ups are joined. Once again, the only join path from A to C is now through B.

Testing Joins

The third tab on the Joins dialog box is the Test tab.

The test feature allows you to test the join strategy connecting tables. The result is displayed in either a diagram format or as an expression. It shows the tables that Impromptu will use when creating reports.

To use the Test tab, in the Available Tables box you select any two tables that could potentially be used in a query, and Add them to the Test Tables list. The Join Strategy box lets you view the series of joins (or lack of joins) that link the selected tables. If there is an error message, you can use the Analyze tab or Edit tab to correct the problem.

For example, if you want to know how a join would be made in a complex report where you are linking many tables, you could test the join first and avoid possible errors or inconsistencies later.

Additional Join Information

This section briefly discusses these additional issues relating to joins:

- Alternate Joins
- Qualifications
- Weighting
- Cross-product joins
- Alternatives to joins

Alternate Joins

Frequently, two non-related tables have a common reference table they both use. For example:

- the Employee table has a Country code that is also in a Country reference table
- the Supplier table has a Country code that is also in the Country reference table

Because of this, a join strategy between these two tables might be established using this reference table as a means of joining the tables. This would be an incorrect join strategy. To correct it, you could add an alias table for the reference table, and change one of the joins for the two tables to the new alias table. This method also applies to loops found in the join strategies for a catalog.

Qualifications

For databases that support qualified tables when building catalogs, qualifications can be set during table inclusion. When catalogs are distributed to your user base, these qualifications might not be applicable. By performing a Qualify Less operation in the Qualifications Tab of the Tables dialog box, such qualifications are removed, allowing user access to the tables.

For more information, see "Qualification Options" (p. 83).

Weighting

In some circumstances, providing "weights" to tables allows Impromptu to optimize queries. The smaller the weight, the earlier the table is added to the FROM clause in the query. For tables that should "drive" most queries, change their weight to a smaller value relative to others.

For more information, see "Table Weighting" (p. 82).

Cross-Product Joins

Cross-product queries are reports that retrieve data from tables with no joins. They take a long time to generate, and can display meaningless results. Since they can hinder performance, Impromptu prevents all user classes from creating cross-product queries. (As an administrator, you can change user class settings to allow a user class to create cross-product joins.)

Where there is no join between tables, a cross-product report is generated. Each row from the table is matched with each row from the other table because there is no specified join condition. These types of reports should be avoided.

Joining multiple tables using any matching columns can result in powerful reporting capabilities, but may have performance costs. You might want to restrict joins to those between primary and foreign keys only. This will limit the flexibility of your joins, but won't incur performance costs.

Alternatives to Joins

When deciding what joins to make, you might want to consider alternatives to some joins. For example, instead of trying to manipulate joins, you might want to create more simple reports that users can drill between, or to create subreports.

For information about drilling through reports and creating subreports, see the *Discovering Impromptu* guide and the online book *Mastering Impromptu Reports*.

For More About Joins

For more information about joins, including step-by-step instructions for creating joins; editing, analyzing, and testing joins; changing joins; and removing joins, see the Impromptu online Help.

Chapter 6: Organizing Folders

The way you organize a catalog's folders can determine how easy it is for users to find information they need. This chapter tells you

- what a folder is
- what you can do with folders
- how to use folders to create a business view of your database

It also includes an example of how you might organize folders as administrator for The Great Outdoors organization, as well as tips and guidelines for organizing your own folders.

What is a Folder?

A folder is a container used to present data in an Impromptu catalog. When you create a catalog, a folder is created automatically for each table in the catalog.

Folders contain data items, which represent columns within a database table. They can also contain items that do not directly reference database columns, such as calculated data items, filter conditions, and report prompts.

One of your key tasks as an administrator is to reorganize a catalog's folders to create a business view that makes sense to your users.

What Can You Do With Folders?

You can use folders to make it easier for users to find and understand the data contained in a catalog. As the catalog creator, you can:

- change the names and organization of folders, to create a more intuitive view of the data they represent
- add new folders that contain subsets of data not directly supported by your database (for example, you could create a folder that contains often-used columns from several tables)
- remove catalog folders that contain unwanted or sensitive information
- create duplicate views of tables and columns within different folders

Your goal should be to organize folders in a way that presents users with one or more business views of your organization's data.

If you completed Chapter 2, "An Impromptu Tutorial," then you have already organized some folders. If you didn't, consider completing the tutorial now, or after reading this chapter. You might also want to experiment with folders in The Great Outdoors catalog (Great Outdoors Sales Data.cat) that comes with Impromptu.

Creating a Business View

When folders are generated during catalog creation, they present a logical view of the database. But the function of folders is to represent a business view of the data, not the database view. As a result, consider the table-oriented view of folders as the starting point for organizing one or more business views of your folders.

What is a Business View?

A business view is a model that represents an organizational structure that mirrors your business and your decision-making processes. Organizing folders into a business view ensures that users who are not familiar with your database structure, and who look for information based on a business-oriented view of the data, can easily locate the data they need.

To present a business view, folders should contain logical groups of information that are typically viewed together. For example, Order Information might span multiple database tables, including Order Header, Order Detail, Products, and Customers. But when users typically view order information they want to see:

Order Header	Order Detail	Products	Customers
Order #	Line #	Product Description	Customer Name
Order Date	Product # Customer # Quantity Price		

You can organize a different business view for each group of users in your organization. A well-designed catalog typically contains multiple business views.

What is Involved in Creating a Business View?

The amount of reorganization needed to create a business view of your folders depends on the structure of your database and the characteristics of your users. It might mean radically reorganizing your folder structure, or it could involve only minimal changes.

If your users are familiar with the database structure, it might be possible to retain much of the folder structure that was originally created from the database table. In this case, you might just want to group details into sub-folders that would be contained by folders of general data. Designing folders according to a business view might result in a folder structure that is similar to the hierarchy of your company's organizational structure.

If your users are not familiar with the database structure, and their business view of the available data is significantly different from the database structure, then you need to put more time and effort into reorganizing your folders. Before starting, carefully analyze exactly what your users' requirements are.

For more information about determining user requirements before administering Impromptu, see "Analyzing User Requirements" (p. 29).

Creating Folders: An Example

This example shows how you might organize folders into a business view as administrator for The Great Outdoors.

The Problem

As administrator for The Great Outdoors, tailoring information delivery for your users has always been a problem. Most report users don't know how the corporate database is organized. As a result, they are lost when they try to create reports from the raw information stored in the database.

For example, while the Great Outdoors database contains tables such as ORDER, ORDRDETL, REP, and CUSTSITE, your senior and regional managers are primarily interested in Sales Orders, Customers, and Products.

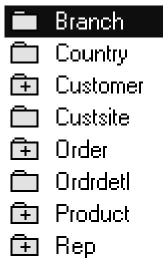
In addition, your management team wants a view of the company's data that highlights sales figures on a global basis, and your regional managers need a folder that highlights sales figures for the countries in their regions.

What's more, both you and your regional managers want to prohibit users from further viewing the details for some fields, such as sales figures for your individual orders, while allowing them access to summary values for that field.

Finally, you must provide your customers with an online catalog of goods that lists your products but denies them access to sales, pricing, and other internal company data.

The Solution

The Great Outdoors catalog contains a single folder for each table in the Outdoors database, as shown here.



To meet the needs of report users, you reorganize the information into a business view that matches your organization's data flow. This involves putting all of the information pertaining to orders in a folder named Orders, all of the fields relating to customers in a folder named Customers, and all of the information for products in a folder named Products.

You also create new folders called Sales by Region, Sales by Country, Sales Year to Date, and Product Brochure. These provide customized views of data for your managers and customers. To make your own job simpler, you create an Admin folder, and place an exact copy of the original catalog folders in it.

Within the folders, you organize information in the way it is used in The Great Outdoors organization. In addition, before users start to create or view reports, you create custom conditions so that only certain data is reported from one user class to the next. You also create custom calculations that enhance the original data in the database.

Within the Customers folder you create a Conditions folder, within which are conditions that filter data for your users. For example, a condition called Active Customers filters out customers who haven't placed orders in the past year. This enables your managers to focus quickly on their active clients.

This new, business-oriented view of your folder structure makes reporting easier for your users, and administering Impromptu easier for you.

Organizing Your Folders

Organizing folders to present a business view of your organization might involve:

- renaming folders and items
- adding folders
- balancing the levels and items in folders
- including calculations, filters (or conditions), and prompts in folders

Renaming Folders and Columns

Folders, and the columns within folders, should have business-appropriate names. You want the names to be understandable both in the catalog and when they are used as the default labels for generated reports. Folder names should be short and concise, to lessen the requirement for scrolling during query generation and viewing.

When organizing your folders, you might end up renaming every single column from the database as a result of this guideline. Remember that table and column names in the database were probably assigned based on the technical naming conventions used by your corporate department of Information Services. Typically, business-appropriate names are different.

To determine appropriate names, work with your users to learn their standard nomenclature, and review existing reports for column and title headings.

Keep in mind that you can use the same column, from the same table, in more than one folder. It can have the same name or a unique name. An example of this might be customer name. This probably comes from the Customers table and might apply everywhere the customer number exists. In this case, every folder might logically contain the Customer Name field.

You might also encounter the reverse situation. For example, suppose you have a Date column name in every table, but it means something different in each case. It might be the Order Date in the Orders table, the Product Release Date in the Products table, and the First Order Date in the Customers table. In this case, to eliminate the possibility of confusion, use different names in the various folders to identify exactly what each date represents.

Adding Folders

When your database does not contain the tables or views that meet your users' needs, you can create folders that display data based on the way your company does business. Simply add the folders you need, then move or copy relevant data into them.

Balancing Levels and Items in Folders

There should be a balance between the number of levels of subfolders in a catalog and the number of items in any one folder. If there are too many items in a folder, users must scroll through lists of items to find what they need. Having too many levels of subfolders increases the time and effort needed for users to navigate folders in search of the data they need for a query.

One suggestion is to insert reference descriptive fields in the same folder as the referenced item or code. For example, add the customer name to folders containing customer ID. This makes query generation simpler.

Including Calculations, Filters, and Prompts in Folders

When creating a business view, ensure that you not only present the information available, but add value to the catalog by building business rules and knowledge into it. One way is to include calculations and filter conditions in folders. This will allow users to view the data in sophisticated ways not supported directly by the database. Another way is to include standard report prompts that users can add to reports they create.

Review existing reports to determine if there are commonly-used calculations, filtering conditions, or prompts that can be better stored in the catalog. They can be stored in the same folder as the database columns, to make it easier for users to find all of the items in one place. You might also want to store conditions, calculations, and prompts in a high-level folder, so that users can find them without navigating through several levels of folders.

Common calculations include Margin, Total Price, Age, and Date calculations, while common filtering conditions include Location and Date filters. Prompts can include any common prompts used in reports to specify data selection criteria, such as those used for date ranges. Ask users to look for common tasks that can be simplified through the use of catalog calculations and conditions, and to identify prompts commonly used in reports.

For more about creating calculations, filtering conditions, and prompts, see "Creating Calculations, Conditions, and Prompts" (p. 63).

Designing Folder Structure vs. User Profiles

You can apply user class folder security to ensure that each user group sees only the folders they need. You can also apply security governors to top-level folders, nested folders, or individual columns within those folders.

The folder structure is one way to present the database to the user and to provide access to the total set of data. Access to the folder structure and individual data items can also be controlled by creating User Profiles that limit users from accessing parts of the folder structure.

For more information about security, see "Setting Up User Classes and Security" (p. 53).

By combining the controls available through User Profiles with those available through folder editing, you can achieve the exact folder presentation that you want for each user community.

Giving Users Control Over Folder Structure

You can assign users the ability to change their own folder structure based on their user profile. The ability is also determined by the type of catalog you are using:

- Distributed. The original folders are locked, but if you allow them to, users can copy these folders in their personal distributed catalog and append new folders.
- Shared and Secured. Users can neither change the structure of the folders nor append to the structure unless you assign them this capability.

Using the Generate Option

If you select the Generate option in the Folder dialog box, the Generate Folder dialog box appears. You can use it to generate a folder with a set of columns and conditions from a query, or to generate a folder containing a table that you want added from the database to your catalog.

Calculations, conditions and prompts can be reproduced in the catalog as long as they do not reference report-specific data items. For example, if a filter condition uses a report data item, the filter is not transferable to the catalog level, but is specific to that report.

If you choose to generate a folder based on a report, Impromptu displays the Select Report dialog box. Once you have selected a report, Impromptu automatically creates a folder with the report name and adds it to your list of folders.

If you choose to generate a folder based on a database table, Impromptu displays the list of database tables. When you select a table, Impromptu creates a folder with the name of the table and adds it to your folders list. If a folder exists with the same name, Impromptu adds a number to distinguish it from the existing folder.

Some Folder Tips

Here are some tips to help you create your own business views of your data:

- Before starting, decide what information is required. Examine existing reports and the reporting objectives of the user community.
- Create the top level of the folder hierarchy first, creating new folders or making copies of existing folders.
- Place folders within folders to create a hierarchical structure consistent with your business data requirements.
- Rename folders or data items, to make their contents easy to identify.
- Create or delete data items, or move or copy them to new locations so users can find them in a way that is intuitive for them.
- Resize the Folders dialog box for a better view of folders in a large catalog.
- Sort the folders in ascending or descending alphabetical order so that users can find them easily.
- Remember to click OK in the Folders dialog box when you have completed the changes. Otherwise the changes aren’t saved.

Some Folder Guidelines

Here are some guidelines to keep in mind when designing folders:

- The number of columns that knowledge workers need to build ad hoc queries is typically between 20 and 75. If you include too many columns in a folder, you might be including information that is not relevant to the user, making it difficult for the user to find information that is relevant.
- Approximately four or five folders should be visible at the top level of the folder structure. Using more than five might limit the ability of the user to get an overall understanding of what data is contained in the catalog and where it is located.
- Keep the hierarchy of folders to three levels to avoid excessive navigation when moving between data items.
- Folders can be used to contain commonly used data items. These items should usually be at the top of the hierarchy so they are easily accessible.
- Store filters in a separate folder or group of folders, so they are easy to locate and can be used with many different reports.

For More About Folders

For more information about folders, including step-by-step instructions for creating, copying, moving, removing, and renaming folders, see the Impromptu online Help.

Chapter 7: Setting Up User Classes and Security

User classes let you create and maintain a single catalog for different user communities, without compromising security. This chapter tells you

- what a user class is
- how to create user classes
- how to set up security with user classes

It includes examples of how you might set up user classes and security as administrator for The Great Outdoors organization.

This chapter also explains how to use Cognos Access Manager for security.

What is a User Class?

If you had to create and maintain a different catalog for each group of users in your organization, your task as an administrator would be huge. But with Impromptu you can define different user classes for different communities who use the same catalog.

A user class is a group of users (or a single user) who need access to the same data in a catalog and who have the same privileges. When you set up user classes, you are creating gateways for different types of users, while retaining a single point of maintenance for yourself.

If you completed Chapter 2, "An Impromptu Tutorial," then you already have experience setting up user classes. If you didn’t, consider completing the tutorial now, or after reading this chapter.

How Do You Create User Classes?

One of your first steps in preparing to administer Impromptu is to determine the groups of users in your organization, the data to which each group needs access, and the capabilities each should be permitted.

You can group users in many ways, including:

- by data requirements (for example, all managers requiring data on employee salaries)
- by departments or divisions within the organization (for example, the European sales region)
- by function (for example, an intermediate level administrator who administers a catalog for a functional group of users)

When you create a catalog, the Creator user class is set up automatically. Users in this class have full control over the catalog. If the catalog you create is a shared, distributed, or secured catalog, an additional class named User is set up as a subset of Creator. The privileges of the users in this class depend on the type of catalog created.

Once you have created a catalog, you can use the User Classes tab of the User Profiles dialog box to define additional user classes. As you create new classes, you can set security by value and by field, and tailor the business view for each class. You can also edit and delete user classes, to accommodate re-organizations, activity changes, and personnel changes.

User classes must be unique for each catalog; you cannot have two classes with the same name. Also, be careful when deleting user classes, because subclasses of the class you delete are also deleted.

Inheritance and User Classes

All user classes except Creator are based on other classes. When you create a new user class, you select the "Parent" class and create a "Child" class based on the parent. A Child cannot have more than one Parent, and it inherits its privileges and restrictions from the Parent.

Setting Up User Classes: An Example

This example shows how you might set up user classes as administrator for The Great Outdoors.

The Problem

As database administrator for The Great Outdoors company, you must provide access to data for:

- senior sales managers
- regional sales managers
- country managers
- sales representatives
- sales outlets (customers)

The reporting needs vary greatly from one class of user to the next. Senior sales managers want concise reports of sales figures on a regular basis, and some of them want the ability to create their own custom reports. The corporate head office needs reports on a company-wide basis, while regional offices want reports for their regions only. As a result, you are constantly running and distributing reports against the same data, filtering out information for each class of user.

The Solution

By looking at the different regions and departments, you can see that there are several classes of report users within The Great Outdoors. Each regional and country office has its own distinct reporting needs, and the sales policies and procedures vary from region to region.

You set up the following user classes to meet these needs:

Sr Sales Mgr Prod Reports—Has access to all data, with several special managerial-level folders set up to highlight pertinent sales trends and reports. This is a secured user class, meaning that report users cannot create or change reports. This meets the requirement for pre-packaged production reports for your senior managers.

Sr Sales Mgr Ad Hoc Reports—Has access to all data, with several special managerial-level folders set up to highlight pertinent sales trends and reports. This user class is not secured, so report users can create new reports or change existing ones. This meets the needs for an ad hoc reporting capability for your senior managers.

Regional Manager—Has access to only the tables, folders, and data values that apply to his or her region. The regional manager acts as a regional report administrator, adding user classes as required and setting up folders to match the specific business view of his or her region. All user classes that a regional manager creates or modifies automatically inherit the security restrictions for that regional manager.

Country Manager—Inherits the restrictions that apply to a regional manager. Each regional manager imposes further restrictions on tables, folders, and data values for the countries within his or her region. As a result, country managers view data that is related to their own country.

Country Sales Rep—Inherits the restrictions that apply to a country manager. Each country manager imposes further restrictions to the data for his or her sales representatives. As a result, each sales representative can see data for his or her customers, and is prohibited from seeing the sales records of other salespeople.

Outlets—Has access only to data about The Great Outdoors products, excluding pricing and sales data. This is a secured user class, meaning that customers cannot alter the reports you provide them. This meets the requirement for electronically distributing an online brochure of your products while maintaining tight security.

The setup for your user classes looks like this:

- **Creator**
  - **Senior Sales Mgr**
    - North American Managers
      - Canada Manager
        - Canada Sales Rep
      - USA Manager
        - USA Sales Rep
      - Mexico Manager
        - Mexico Sales Rep
    - European Managers
      - UK Manager
        - UK Sales Rep
      - Germany Manager
        - Germany Sales Rep
      - France Manager
        - France Sales Rep
      - Sweden Manager
        - Sweden Sales Rep
    - Orient/Australia Managers
      - Japan Manager
        - Japan Sales Rep
      - Hong Kong Manager
        - Hong Kong Sales Rep
      - Singapore Manager
        - Singapore Sales Rep
      - Australia Manager
        - Australia Sales Rep
  - **Outlets**
    - North American Outlets
    - European Outlets
    - Orient/Australian Outlets

Once you create the user classes that meet the needs of your corporate user community, you can tailor the ways these user classes can access and view data.

How to Create User Classes

You create user classes using the User Classes tab of the User Profiles dialog box.

Instructions for creating user classes are included in the online Help. If you want practice creating user classes before you start creating your own user classes, complete Chapter 2, "An Impromptu Tutorial."

User Classes and Security

There are a number of ways you can employ user classes to set up security for your database. You can:

- assign passwords to user classes
- limit table access by user class (field-level security)
- limit folder access and restrict select values by user class (folder-level security)
- filter values by user class (security by value)
- set governor values by user class
- set database tab values by user class

As you create user classes below other user classes, the security settings are inherited. The combination of user class and inherited security means you can create a single catalog and a few reports. Impromptu automatically channels the right data to the right people, based on your security settings.

Assigning Passwords

When you set up a user class, you can assign it a password. Users in that class will be prompted for the password when they attempt to use a catalog. To change or remove a password, you edit the class definition.

If a user forgets the user class password, and for some reason you have no record of it, you as the Creator can create a new password. But if you forget the Creator password you cannot create a new one; you must recreate the catalog.

Impromptu's database access features also enable you to store database user IDs and passwords in the catalog. Alternatively, if you don't want to store this information in the catalog, you can allow Impromptu to prompt your users for their database user IDs each time they run reports. Impromptu stores user IDs and passwords in an encrypted format in the catalog.

Limiting Table Access

Using the Table Access tab of the User Profiles dialog box, you can control access to tables.

Table-level Security

Table access can be granted or denied for each user class. If access is denied at the parent level of a class hierarchy, then all child classes below that parent are automatically also denied access. (The table does not appear in the list of the child class.) When access to a table is denied, any reference to that table in a report will result in a query execution error.

You can also give a user class access to a table but not to a folder. Users in the class can then run a report that uses the table, but cannot change the report.

Notes

- You can control data access to particular table columns or to the entire table. When you control access to the entire table, access is prohibited to all of the table's columns.
- Specifying table/column access using Table Access overrides and filters down to Folder-level Security.

Limiting Folder Access and Restricting Select Values

Using the Folder Access tab of the User Profiles dialog box, you can control data access through the catalog's folder structure. You can also increase performance by restricting the ability to perform the Expression Editor's Select Values function on desired folder items.

Notes

- You can control data access through the folder structure or the Select Values function to particular folder items or to entire folders. If you restrict the ability to perform a Select Values on desired folder items, all items in the folder are affected.
- A database table column may appear more than once in the catalog's folder structure, and under different names.

Folder-level Security

Folder access can be granted or denied for each user class. If access is denied at the parent level of a class hierarchy, all child classes below that parent are automatically also denied access. When folder access is denied, the user does not see that folder when using the catalog to create or modify reports.

A folder structure is often organized by a business view, group, or perspective. Security by business view prevents users from seeing folders that contain sensitive data they are not allowed to see. For example, you can restrict members of a user class from accessing data from another department.

Table-level vs. Folder-level Security

One reason for separating table and folder access is to deny access to a folder for ad hoc reporting, but allow access to its contents for standard reports. The following table outlines the difference between denying access at the table level versus the folder level:

If ...	and ...	then ...
you deny access to a column in a folder or to the entire folder	you grant access to the column in the table or to the entire table	users can run existing reports, but cannot create a report with that column/folder because it does not appear in Catalog Data
you grant access to a column in a folder or to the entire folder	you deny access to the column in the table or to the entire table	users cannot run existing reports (they get an error message) and cannot create reports using this column or table
you either grant or deny access to a column in a folder or to the entire folder	you deny access to a table which is needed for an intermediate join	users can still execute a query that includes columns from the two tables which are joined via this denied table. The Deny feature in Impromptu restricts what is displayed to the members of a user class but does not prevent the joining of the tables.

Restricting Select Values

By denying the ability to perform a Select Values, you increase performance by reducing the resource consumption required to load a large list of values that may or may not be indexed.

Filtering Values by User Class

Using the Filters tab of the User Profiles dialog box, you can set up conditional expressions that will filter out unnecessary or sensitive data for a user class, giving you security by value.

You can create a user class filter for a table or column. It is automatically applied when users in the user class access that table or column.

A filter retrieves a specific set of records for a user class, and is automatically applied when a user accesses the table or column associated with the filter.

Filters defined in the User Profiles dialog box apply to the class they are associated with, and to any of its child classes. Filters allow you to specify that only certain rows (or records) are accessible to the user. The filter is automatically applied only when the table or column it is associated with is included in a report.

For hands-on experience creating filters, complete Chapter 2, "An Impromptu Tutorial." For more information about creating filters, see "Creating Calculations, Conditions, and Prompts" (p. 63).

Setting Governor Values

Settings in the Governor tab of the User Profiles dialog box enable you to apply limits on processing capabilities by user class.

The first block of settings—Sorting on Non-Indexed Columns through Cross-Product Queries (No Table Joins)—permit you to Allow, Warn, or Prevent various options. The warn and prevent message will appear when Impromptu detects the specified condition.

The second block of settings determines whether the user can perform the specified action. These include

- the Create/Edit Reports checkbox
- the Edit Folders checkbox
- the Add/Modify User Classes checkbox
- the Direct Entry SQL checkbox.

Create/Edit Reports

If the Create/edit Reports box is selected, users in the user class can create, edit and run reports. If it is cleared, the users can only run reports. Use of this checkbox is appropriate if the user class will be doing ad hoc reporting. If they will only execute standard reports, clear this checkbox option (this is what the secured catalog does). Clear this box when you are changing an existing catalog to a secured catalog.

Edit Folders

If the Edit Folders checkbox is selected, users in the user class can append folders to the catalog structure and modify the folders. This is appropriate if the user class has the ability to modify the folders, as is the case with personal and distributed catalogs. For shared and secured catalogs, this checkbox is cleared.

Add/Modify User Classes

If this option is selected, users in the user class can create new user classes. This means the users can share some of the Impromptu administrator tasks. For example, you can allow the Managers user class to create user classes for the Employees they manage. This option will only work for users who have the Administrator version of Impromptu.

Direct Entry SQL

If this option is selected, users in the user class can edit Structured Query Language (SQL) in reports, and the Edit SQL button will be available in the Query dialog box. This is appropriate only for users who have a strong knowledge of SQL. Also, it reduces the portability of the catalog. In most cases this checkbox should be cleared.

Using the Database Tab

The Database tab of the User Profiles dialog box is available only for databases that support certain options. If the tab is available, you can use it to:

- set up a prompt for a password when a user class attempts to access the database (does not apply to all databases)
- predefine database logon information (does not apply to all databases)
- set the transaction isolation level supported by your database (You should always attempt to use the lowest isolation level possible to minimize the potential impact of long running queries on data that might be concurrently updated. Please refer to your vendor documentation for more details on transactions and isolation levels.)

Using the Prompt for Database Security option, you can have Impromptu prompt for a database user ID and password when users open a catalog.

Alternatively, you can associate a database user ID and password with an Impromptu user class. This allows users to access a database without knowing what user name or password is being used. If the database server has enabled password aging, and a password has expired, the catalog entry must be updated before the user can connect to the database again.

If the password is embedded in the catalog, make sure that password aging is not applied. If password aging is applied, when a password expires the user will be unable to define a new password and unable to access the catalog. If you use password aging, place the password in the Database account.

In addition to setting login standards, you might be able to set the transaction isolation level supported by your database. The available levels might include any of read uncommitted, read committed, cursor stability, reproducible read, phantom protection, and serialization. Not all databases support all levels of transaction isolation. Check with your database administrator or database vendor for more information.

User Classes and Catalog Types

When you create a new catalog, you choose the catalog type that establishes the initial configuration of the Governor settings.

For example, if you initially create a personal catalog, you can later add user classes and change the Governor options. Depending on the changes you make, the catalog might be secured for some users (you deny the Create/Edit Reports governor and the Edit Folders governor for one or more user classes) and shared for others (you permit the Create/Edit Reports governor but deny the Edit Folders governor for one or more user classes).

In addition, you can select the Make This a Distributed Catalog attribute in the Catalog Properties dialog box to make any catalog distributed, so that users work with their own individual copies of the catalog regardless of their privileges and restrictions, and changes to the source catalog are propagated to each user's copy.

The catalog type is dependent upon what you want your users to be able to do; that is, how you define the user profiles.

Note that there is no checkbox for changing from one catalog type to another, except for making a catalog distributed. Instead, you can choose the options (privileges) that you want in the Governor tab. You can change any type of catalog into any other type of catalog by modifying the Governor options for any user class.

This table summarizes the privileges associated with catalog types:

	Run Reports	Create/Edit Reports	Edit Folders	User Has Own Copy	Meant for Multiple Users
Personal	✔	✔	✔		
Secured	✔				✔

	Run Reports	Create/Edit Reports	Edit Folders	User Has Own Copy	Meant for Multiple Users
Shared	✔	✔			✔
Distributed	✔	✔	✔	✔	✔

Establishing Security: An Example

This example shows how you might establish security as administrator for The Great Outdoors company.

The Problem

As the database administrator, one of your biggest problems in information distribution is security. Each user class requires access to specific data, and you must secure data so that it is accessible only by the people with authority to see it. As a result, you are constantly running different reports against the same data, with each new report tailored to meet specific security requirements.

For example, although the regional managers require regular reports about sales activity within their own regions, you don't want them to see sales figures from other regions. Your European manager does not want to see North American sales figures. Similarly, sales representatives must be restricted in the way they can access and view sales and quota figures. Most important, the online product brochure that you provide your customers must be tightly restricted to data for your company's products, excluding your internal pricing information.

The Solution

To handle security requirements for data in the Great Outdoors database, you use security by field, security by folder, and security by value.

In order to secure product sales information from your customers, you set security by field so that the user class Outlets can see only the product information you want them to see. In this case, you want to grant them access to the following information:

- product number
- product name
- product type and line
- product picture

To do this, you deny the Outlets user class access to all tables but Products.

Within the Products table, you deny the Outlets user class access to all columns except Prod_No, Product, Prod_Type, Prod_Line, and Picture.

You also decide to ensure that sales outlets are not only restricted by field, but that they can not see beyond the Products folder. To do this, you impose security by folder on the Outlets user class so its members are denied access to all folders but Products. In this way, you not only prevent outlets from viewing data, but also from learning about how the data is organized.

To secure data for your regional managers, you set up filters that grant them access to their own regional data only. For example, you set a filter for the European Region manager that limits his or her view of countries to Europe only.

As you set security for user classes, Impromptu automatically ensures that subordinate user classes inherit security restrictions from the user class above them in the class hierarchy. In this way, Impromptu automatically ensures that individual salespeople are automatically restricted to the data that is accessible by their country manager. In turn, the country manager is automatically restricted to data that is accessible by their regional manager, and so on. Once your regional managers receive the catalog, they can impose further restrictions on the user classes below them (country manager and country sales rep).

To deny everyone but yourself access to the original catalog folders, you set security by business view. Recall that when you reorganized the business view for the Great Outdoors catalog (see "Creating Folders: An Example" in Chapter 6, "Organizing Folders"), you created an Admin folder in which you placed a copy of the original folders as generated from the database. By denying your senior managers and outlets access to this folder, you automatically ensure that no one but you can see the contents of the Admin folder.

Securing Access to the Database: An Example

This example shows how, as administrator of The Great Outdoors, you might overcome the need for users to remember multiple IDs and passwords.

The Problem

Some of your report users complain about having to remember user IDs and passwords when they open reports. In particular, members of your senior management team always work in secured areas that no other users have access to, so they don't want to be bothered with user IDs and passwords. In addition, your customers often have difficulty keeping up-to-date with password changes for the corporate product brochure that you provide them. For this reason, managers and customers have asked that the user ID and password prompts be suppressed.

The Solution

To remove the necessity for your managers to remember multiple user IDs and passwords for the information they need to access, you set the Senior Sales Manager user class so that it automatically attaches to the database with a specific user ID and password that Impromptu stores in the catalog. You do the same thing for customers who need access to the online company product brochure.

The Great Outdoors database is updated by sales staff as new orders are placed. On an average day, this can mean a few thousand transactions. You decide to allow managers to view data that is currently being updated by another user. This gives your sales managers the most up to date information possible.

Using Access Manager for Security

Impromptu comes with Access Manager, a set of tools you can use to control security for all your Cognos products. You may want to use Access Manager with Impromptu catalogs to avoid forcing users to log on more than once when they move between applications. Access Manager also allows you to administer security for all your catalogs in one place instead of maintaining separate security structures for each catalog.

Access Manager uses LDAP technology: a list of users and user classes and the corresponding passwords and permissions are stored in a namespace on a directory server. When you log on to a Cognos product, Access Manager uses this information to determine whether you have the appropriate access privileges.

To integrate with Access Manager, you must ensure that

- Access Manager is installed and configured properly on your computer

  This includes specifying a default namespace for your computer. For more information, see the Access Manager documentation.
- the user classes in your catalogs match the user classes in your Access Manager namespace

When you open a catalog, Access Manager prompts you for signon information and checks the namespace. If Access Manager finds your signon in the namespace, it determines which user classes you belong to and checks the catalog. If you belong to one user class that exists in the catalog, Access Manager logs you on using that user class. If you belong to more than one user class that exists in the catalog, Access Manager prompts you to pick a user class, then opens the catalog.

If Access Manager can't determine who you are, or if you don't belong to a user class that exists in the catalog, you can still log on using catalog security. However, if you don't want to provide your users with the option of using catalog security, you can type a namespace in the Catalog Properties dialog box (Catalog menu). If you can't log on using that namespace, or if you don't belong to a user class that exists in the catalog, Impromptu denies access to the catalog.

If Access Manager can find all the information it needs, you can log on to the catalog without typing a signon. Clear the Always Show This Dialog check box (Catalog Logon dialog box) to log on without typing a signon. To show the Catalog Logon dialog box again, from the Catalog menu click Disconnect, then ctrl-click Connect.

Access Manager and User Classes

One of the benefits of Access Manager is that it can maintain a single user class structure for all your catalogs. If you use Access Manager to control security for more than one catalog, you must ensure that the user classes in all your catalogs are synchronized.

For example, Catalog1.cat contains a user class called Sales that regional sales administrators use to track the performance of each sales representative. Catalog2.cat contains a user class called Sales that sales representatives use to check their own performance against quota. If you move both Sales user classes to a single Access Manager namespace, sales representatives will gain access to each other's performance information. Before you integrate with Access Manager, you should rename the user classes to Sales Admin and Sales Rep.

Also, each catalog automatically contains a Creator user class. If you add a user to the Creator user class in Access Manager, that user will have access to all your catalogs.

For More About User Classes and Security

For more information about user classes and security, see the Impromptu online Help. From the Help menu, click Contents and Index. Then, in the Contents tab, select:

- User Classes and Privileges
- Control the User Environment

For more information about using Access Manager with Impromptu, see the Impromptu Administrator online Help.

Chapter 8: Creating Calculations, Conditions, and Prompts

Creating calculations, conditions, and prompts in a catalog can ensure reporting efficiency and consistency throughout your organization. This chapter tells you

- what calculations, conditions, and prompts are
- why you might want to create calculations, conditions, and prompts
- which calculations, conditions, and prompts you might want to create
- where to store calculations, conditions, and prompts
- how to create calculations, conditions, and prompts

This chapter also includes examples of calculations, conditions, and prompts used by The Great Outdoors organization.

What Are Calculations, Conditions, and Prompts?

With Impromptu, you can define calculations, conditions, and prompts, and include them in catalogs.

Calculations

A calculation is an expression that you define to perform operations on existing data items to create a new value. For example, this calculation uses the existing items of Price and Cost to calculate a value for Low Margin:

```
if ((Price - Prod Cost) <= 0.20 * Price) then (‘Low Margin’) else NULL
```

Conditions

A condition (also called a filter or a filtering condition) is a business rule or calculation that evaluates to a true-false value. For example, this condition uses a table’s Country column to filter data so that only North American data is included in a report:

```
Country in (‘United States’, ‘Canada’, ‘Mexico’)
```

Prompts

Prompts are included in reports so that users can specify selection criteria when generating the reports. For example, the prompts used to specify a date range for the data to be included in a report might be:

```
Enter the Start Date of the date range
Enter the End Date of the date range
```

You might recognize the above calculation, filter condition, and prompts as the ones created in Chapter 2, "An Impromptu Tutorial." If you skipped the tutorial, and you want hands-on experience creating a calculation, condition, or prompt, consider completing the tutorial after reading this chapter.

Why Create Calculations, Conditions, and Prompts?

Creating calculations, conditions, and prompts, and storing them in a catalog, can have several advantages:

- They save time. You and your users don't have to re-create them every time you need them. For example, defining a Low Margin calculation once means users do not have to define it every time they use the low margin value in a report.
- They ensure consistency. Users throughout your business can use the same calculations, conditions, and prompts. For example, the same Low Margin calculation would be used by all your users, ensuring that low margin figures always mean the same thing.
- They are easy to update. You can maintain calculations, conditions, and prompts centrally, so all reports are updated automatically as calculations, conditions, and prompts evolve and new ones are added. For example, if you change the calculation for Low Margin, all reports that use the Low Margin calculation are updated automatically.
- They enhance security. You can use filters to limit the data that certain user classes can see. (For an example, see "Filtering Values by User Class" in Chapter 7, "Setting Up User Classes and Security.")

Which Calculations, Conditions, and Prompts Should You Create?

You can create calculations, conditions, and prompts as the need arises. But you might want to anticipate the need for them when you are building your catalog.

For example, review existing reports to determine if they have commonly-used calculations, filtering conditions, or selection criterion prompts that you can create once and store in the catalog. Common calculations might include Margin, Total Price, Age, and Date calculations. Common condition requirements might include Location filtering and Date filtering. And common prompts might relate to time ranges or account numbers.

One way to determine common calculations and conditions is to ask users to look for common tasks that can be simplified using catalog expressions and conditions. Similarly, you can review existing reports to determine commonly used prompts that you want defined in the catalog.

You can create calculations, conditions, and prompts that users can include as components of their own calculations, conditions, and prompts.

Storing Calculations, Conditions, and Prompts

Calculations can be stored next to the data items used to create them. For example:

- The calculation for Profit could be stored with Sales Revenue and Expenses.
- The calculation for Sales Growth could be stored with Sales 97 and Sales 96 data.

Conditions can also be stored in separate folders. Some examples of separate folders that can be used to store conditions are:

- Time periods. These could contain conditions for fiscal years, quarters, YTD, QTD, Year-over-Year, current month Year-over-Year, and so on.
- Geographic. These could contain conditions for individual locations, or geographic reporting roll-ups.
- Functional. These could contain summary filters on levels of the product hierarchy or distribution channels.

You can store all prompts in a separate Prompts folder, or store individual prompts in folders containing data items relating to the prompts. For example:

- Prompts related to the types of sales could be stored in Sales folders
- Prompts used to specify the locations to include when retrieving data could be stored in Geographic folders

Creating Calculations, Conditions, and Prompts: An Example

This example describes conditions, calculations and prompts you might define as administrator for The Great Outdoors.

The Problem

One of the complaints you have received about reports over the years is that there aren't enough standards. Often reports do not agree from one region to the next. For example, a calculation of year-to-date sales and profit margins from Europe is calculated differently than the same figure for North America. This occurs because the managers for these regions use their own calculations to derive these values. As a result, a comparison of profit margins between the two regions is difficult.

The same problem holds for conditions. For example, when preparing a report of sales figures for European and North American regions, it is important that all sales managers use the same definitions for European and North American customers. If one sales manager forgets to include sales from a particular country, his or her sales reports will contain incomplete results.

Finally, because users throughout The Great Outdoors have defined their own report prompts, different prompts are used for the same purpose in different reports, which could cause confusion among the reports' users.

The Solution

In the Great Outdoors catalog, you define the following catalog conditions to ensure that customers are always grouped correctly:

- North American Customer, where the condition is
  Country Code in ('CAN' , 'USA' , 'MEX')
- European Customer, where the condition is
  Country Code in ('UK' , 'SWE' , 'FRA' , 'SPA' , 'BEL')
- Orient/Aus Customer, where the condition is
  Country Code in ('JAP' , 'HKG' , 'SING' , 'AUS')

Similarly, to track levels of sales representative performance across regions, you create catalog conditions that isolate normal sales versus those that didn't close within a specified number of days:

- Sale Closed On Time, where the condition is
  Day(Closed Date - Order Date) <= 5
- Sale Closed Late, where the condition is
  Day(Closed Date - Order Date) > 5

To ensure the consistent use of selection criteria prompts, you define the most commonly used prompts in your catalog, including:

- From Date and To Date for date ranges
- Cost Center, Department, Location, and other prompts for geographical and organizational components of your organization

There is virtually no limit to the customization you can add at the catalog level for your report users.

How to Create Calculations and Conditions

To start a new calculation or condition, you click the New Calculation or New Condition button in the Folders dialog box to display the New Calculation or New Condition dialog box.

Both the New Calculation and New Condition dialog boxes provide:

- a Name box, to assign your calculation or condition a name
- an Expression box, to create and edit the expression that defines your calculation or condition
- an Available Components box, which contains the components you can use in the expression
- a Tips box (at the lower right), where Impromptu displays helpful hints and error messages relevant to the expression you are creating

Building Expressions

To create a calculation or condition, you define an expression in the Expression box using the syntax rules enforced by Impromptu. You select expression components from the Available Components box. The box shows only those components that make sense in the given expression context.

To move around in the Available Components and Expression boxes you can use the mouse, or:

- press the space bar to select a component
- press the up and down arrows to move within the Available Components box
- press the left and right arrows to move within the Expression box
- press the Enter key to commit a value within an expression and move the cursor to the right of the value

The components of an expression can include: Functions, Summaries, Values, Catalog Columns, and Operators.

Functions

The Functions folder in the Available Components box contains functions provided by Impromptu or by your database. These are pre-defined calculations, such as string, numeric, and date calculations, that are designed to operate on various data types.

Summaries

Summaries calculate additional data items by applying a mathematical procedure to selected data items in a report or catalog. They include:

- Average: Returns the average value of selected data items
- Count: Returns the number of selected data items
- Maximum: Returns the maximum value of selected data items
- Minimum: Returns the minimum value of selected data items
- Percentage: Returns the percent of the total value for selected data items
- Percentile: Returns the percentile value of selected data items
- Rank: Returns the rank value of selected data items
- Running-Average: Returns the running average value of selected data items
- Running-Count: Returns the running count of selected data items
- Running-Maximum: Returns the running maximum of selected data items
- Running-Minimum: Returns the running minimum of selected data items
- Running-Total: Returns the running total of selected data items
- Total: Returns the total of selected data items
- Standard Deviation: Returns the standard deviation of selected items

Summary results depend on the set of data that is summarized. For example, if a column of data is grouped in your report, then a different result will be created for each of those groups from the single application of a summary.

Values

Values include string, number, date, time, date-time, and interval. They can be entered into the expression in three ways:

- From the Value folder as a string, numeric or data. Values are typed directly into the expression once the type of value is determined. Remember to press Enter to complete the entry.
- From the catalog data, by referencing database columns directly.
- From the report data, by referencing items already in the report.

The Expression editor is context-sensitive, so it presents only options that are relevant at each stage of expression building.

To apply a catalog filter to a report, select the Filter tab of the Query dialog box, then choose the catalog filter from the catalog data options, then add it to the Expression box.

Operators

Operators specify what happens to the values on either side of the operator. There are four types:

- logical operators define a relationship between two parts of an expression (for example: and, not, or)
- arithmetic operators perform arithmetic operations on two parts of an expression (for example: +, -, *, /)
- string operators concatenate two character strings (+)
- comparison operators filter data by comparing one or more values that you enter against the values in the database (for example: like, =)

For more information about creating expressions, see the Impromptu online Help. From the Help menu, click Contents and Index. Then, in the Contents tab, select "Impromptu User", then select "Operators and Other Components" or "Functions."

Some Tips for Creating Expressions

If you complete Chapter 2, "An Impromptu Tutorial," you will get hands on practice creating expressions. If you haven’t completed the tutorial, you might want to complete it now, or after reading this chapter.

Here are some additional tips to help you create calculation and condition expressions:

- When you select String, Impromptu inserts two quotation marks and positions the cursor between them. You enter the string between the quotation marks.
- When you select a number, Impromptu inserts the number 0. You overwrite it with the number you want included in the expression.
- When you select date-time values, Impromptu inserts the current date and/or system time in quotation marks. You can overwrite it with other data values.

How to Create Prompts

To start a new prompt, you click the New Prompt button in the Folders dialog box to display the Prompt Definition dialog box.

In the dialog box, you can

- name the prompt
- specify its type (Type In, or one of three varieties of Picklists)
- enter the prompt message to display
- specify the data type and default value

When you click OK in the Prompt Definition dialog box, the prompt you have defined is displayed in the Folders dialog box. A question mark icon beside it indicates that the object is a prompt. For example:

Start Date

Users can use catalog prompts in reports but cannot change or delete the prompts. When a user includes a catalog prompt in a report, the prompt becomes available in the Prompt Manager.

For more information about the Prompt Manager, see *Discovering Impromptu* or the online book *Mastering Impromptu Reports*.

For More About Calculations, Conditions, and Prompts

For more information about calculations, conditions, and prompts, see the Impromptu online Help. From the Help menu, click Contents and Index. Then, in the Contents tab, select

- Work with Conditions, Calculations and Prompts

Chapter 9: Automating and Updating Impromptu

This chapter describes features to help you automate and maintain your Impromptu environment, including

- standard reports and templates
- the inheritance feature
- the Impromptu.ini file
- Impromptu Macro Editor
- Impromptu Dialog Editor
- Cognos Scheduler
- the Impromptu Type Library

Overview: Six Ways to Automate Impromptu

Impromptu has features to help you automate and maintain your Impromptu environment. You can:

- create standard reports and templates
- take advantage of Impromptu's inheritance feature
- distribute a revised Impromptu.ini file to users
- mail shortcuts
- automate tasks with CognosScript Editor, CognosScript Dialog Editor, and Cognos Scheduler
- use the Impromptu Type Library

Standardizing Reports and Templates

One way to make your job as an administrator easier is to create standard reports and templates for your users. These help ensure consistency, and make it easier for you to maintain Impromptu and for your users to use Impromptu.

Standard Reports

A standard or production report is a report that you create and distribute to users. You can do this by creating

- a secured catalog where users cannot create or alter reports
- a distributed or shared catalog and placing the associated reports in a read-only directory. This prevents users from modifying the original copies of the production reports that you distribute, but allows them to make changes to their own copies of the reports.

Creating standard reports gives users an easy entry point into Impromptu, and ensures that your reports have a consistent look and feel.

Creating Reports That Users Can't Change

You might want to prevent users from modifying certain standardized reports. For example, customer statements, invoices, and monthly profit and loss reports should reflect company standards, and should not be modified by users.

You could create a specific catalog for these reports, but that would increase your maintenance time. Instead, try making a read-only file or directory that cannot be modified regardless of the type of catalog used. Users can open the report/template and save a report using a new name.

Templates

Templates are patterns or models that can be used to build reports. Like standard reports, templates help ensure report consistency throughout your organization. They give users a starting point for creating reports, but don't limit users to the extent that standard reports do.

Templates are the basis for all reports. They can be as simple as the default simple list template supplied with Impromptu. They can also be much more complex, and include place holders, calculations, prompts, and formatting.

Impromptu comes with several templates that you can use to begin creating your own business reports. If your users create the same type of reports frequently, you might also want to provide them with a set of standard templates you have created. You can use the Impromptu default templates as is, modify them to suit your business requirements, or create your own templates.

Templates can be used for classic reporting styles such as mailing labels, invoices, sales reports, expense reports, name and address books, and product catalogs. You can roll out standard templates with your Impromptu catalogs and have users access them immediately.

The possibilities are endless when designing templates. For example, you can define two templates with similar headings but different page orientation (portrait and landscape). Or, you can use a single template and allow the user to switch the orientation as needed. To keep the proper alignment of headings with both paper orientations, use the alignment options Center, Left, and Right Aligned With Parent. Impromptu automatically adjusts to changes in page size and paper orientation.

If you include placeholders in templates, to represent data items that should appear in the report, you make it easy for users to know the type of data to include in reports built from the template. Mailing labels, for example, might include placeholders for name and address.

For information about creating templates, see the online book *Mastering Impromptu Reports*.

Providing Standard Reports and Templates: An Example

This example explains how The Great Outdoors uses standard reports and templates to ensure consist reporting throughout their organization.

The Problem

As a world class company, it is important that you provide reports with a consistent look and feel for all of your Great Outdoors users. On the other hand, because The Great Outdoors company does span the globe, it is equally important that you give regional report users the ability to customize their reports to meet local requirements. In addition to controlling the look of reports, you must provide your users with "head starts" that enable them to build their own reports quickly and easily.

The Solution

To satisfy the common needs of your users, you create a set of standard templates and reports. For example, for senior, regional, and country managers, you create a template called Top Performers. Within this template are placeholders into which managers can place key sales data items, such as sales, product, and sales representative. The template automatically groups the data by the data items against which sales are measured, and then applies ranking and conditional highlighting to the report values so that key performers stand out immediately. Because you have already set up security by value for your regional and country managers, they automatically see the top performers within their own regions.

For your salespeople, you create standard reports that outline their individual sales performance. In addition, you create both templates and standard reports that enable salespeople to browse customer and product information. The templates enable sales personnel to create their own customer and product reports, while the standard reports give all sales representatives the same kind of information in a consistent format.

For your customers, you create a standard product brochure report. This report displays product information that your customers can use in their own promotions and sales outlets. Because you earlier secured the user class Outlets by denying its users the ability to create or change reports, your customers cannot change the information in your product brochure. They can use it in sales situations at any time.

How Inheritance Helps You Update and Maintain Impromptu

Impromptu's powerful inheritance features enable you to easily maintain and update the reports you distribute. Because all user classes are subordinate to yours, you control what each user class can access. Moving down the user class hierarchy, each user class automatically inherits the security restrictions of the user class above it. As a result, when you make a change near the top of the user class hierarchy, the change occurs throughout your entire user community.

The inheritance of higher-level settings applies to:

- standard conditions, calculations, and prompts in the catalog. If your users have added a standard condition or calculation to a report, and you change the condition or calculation, the change is automatically reflected in the next report created.
- changes in table or folder access
- changes in the database or the business view

Making changes in any of these areas causes the changes to automatically apply to your users. In the case of a shared catalog, all users access the information in a single catalog. Changing that catalog ensures that each user sees the changes the next time he or she opens that catalog. In the case of a distributed catalog, changes are inherited when the catalog distributes itself (makes a personal copy), or a personal copy is updated.

Updating and Maintaining Your Environment: An Example

This example shows how The Great Outdoors responds to the need to update and maintain their Impromptu environment.

The Problem

The original creation of a reporting solution for The Great Outdoors company is actually a small part of your job as the database administrator. As processes and operations change within the company, you constantly receive requests for new reports. In addition, as the database changes over time, you are continually performing time-consuming maintenance on the reports that you distribute.

For example, in a single day, you receive memos that inform you of the following company activities:

- European operations are changing effective one week from today, with the addition of new offices in both Greece and Holland.
- For security considerations, product sales history data is no longer to be distributed beyond the regional manager level.
- A major new account has been signed with a company named Total Fitness Shops. This new customer has over 40 retail outlets throughout the United States.

The Solution

After you update your corporate database to accommodate the new regional offices in Europe, and you add the data for the new account, updating the catalog to match is simple.

To handle the new country offices in Europe, you enhance your European Customers condition so that it contains the new country codes for Greece and Holland.

You could make this task even simpler if you store your country codes in a dataset and update the report that creates the dataset.

To enforce the new security restrictions, you deny access to sales history data to all country managers. Impromptu's inheritance feature ensures that sales personnel are automatically restricted in the same fashion.

To make the users' views of data match the new business view, you add a custom condition called Total Fitness Outlets to your catalog. This provides instant access to information about the new customer to all of your American managers and salespeople.

Automating Tasks with Macros

A macro is a set of instructions that enables you to automate complex or repetitive tasks. You can use macros to:

- generate reports
- use existing library functions and subroutines or ones that you create
- perform complex string and numeric operations
- open and close files
- activate other applications
- generate snapshots and HotFiles
- integrate applications using OLE

As part of your overall distribution strategy, you can use Cognos Scheduler, together with Impromptu's advanced OLE automation methods and the CognosScript macro language. By combining these powerful utilities, you can automate the distribution of Impromptu reports while integrating Impromptu with other applications at the same time.

Cognos Macro Utilities

The three main utilities used to create, test, debug, and schedule Impromptu macros are:

- CognosScript Editor
- CognosScript Dialog Editor
- Cognos Scheduler

When you install Impromptu, these utilities are placed by default in the same program group as Impromptu.

CognosScript Editor

You can use CognosScript Editor to create, test, and debug macros. Use this utility to enter the macro commands using the CognosScript language. You can run CognosScript Editor from within Impromptu or directly from Windows.

You can use CognosScript Editor to:

- open, close, save and rename macros
- access the dialog editor
- create, copy, cut, undo, and find text
- check syntax (debugging)
- run macros
- debug or single-step through instructions
- set/clear breakpoints
- display functions

CognosScript Editor and OLE Automation

Among other things, you can use CognosScript and OLE automation to:

- create, read, update, and delete catalogs
- create and maintain joins
- write and edit expressions
- create and maintain user classes
- manage database connections

For more information about CognosScript Editor, refer to the CognosScript Editor online Help.

CognosScript Dialog Editor

You can use CognosScript Dialog Editor to create custom dialog boxes for use in your more complex Impromptu macros. You can run this utility from within Impromptu or directly from Windows.

For more information about CognosScript Dialog Editor, refer to the Dialog Editor online Help.

Cognos Scheduler

Cognos Scheduler launches reports, programs, and macros based on times and dates that you specify. These scheduled launches are called "tasks."

For example, you can create a task that launches:

- a CognosScript macro that automatically runs a production report or builds a HotFile every day of the week, starting at 2:00 A.M.
- an Excel macro to run once a week on a specified day at a specified time. Your Excel macro might include OLE automation statements that automatically run a linked Impromptu report.
- a CognosScript macro that automatically runs a set of Impromptu reports once a month, then mails the reports to a group of users
- a request to batch Impromptu reports on the database server or Request Server
- a request to schedule an Impromptu query with the Request Server

Scheduler enables you to run programs after hours, freeing you for other work during the day and reducing network traffic during peak hours.

Use Cognos Scheduler to run automated processes at pre-set intervals, whether or not you are at your computer. You can use Cognos Scheduler to run tasks unattended, since you can specify security parameters for your databases and Impromptu catalogs within the Scheduler. Cognos Scheduler stores catalog user IDs and passwords in an encrypted format, so they are not visible when a task is running.

Once you have created a task and scheduled it to run at a specific time, all you have to do is leave your computer on with Cognos Scheduler running. (If you are working on the Request Server, this is not necessary.) At the specified time, Cognos Scheduler will automatically perform the tasks.

To have Cognos Scheduler working in the background, minimize it or run it from the Windows application tray on the taskbar.

Starting Cognos Scheduler

You can start Cognos Scheduler:

- from the Cognos folder (the recommended method)
- from the Start button in Windows
- from the Windows Explorer by double-clicking Schdl_go.exe
- from Impromptu by choosing the Scheduler command from the Tools menu
- from Impromptu by clicking the Launch Scheduler button on the toolbar

To have Cognos Scheduler run when you start Windows, copy the shortcut to the startup folder.

For more information about Cognos Scheduler, refer to the Scheduler online Help.

The Impromptu Type Library

Impromptu comes with a type library that contains descriptions of all the exposed Impromptu objects, properties, and methods. The type library makes it easier for programmers in C++ to use Impromptu automation components in their code.

To access the type library, reference the Impclient.tlb file in the folder where Impromptu is installed.

For more information about the type library, see the Impromptu Administrator online Help.

For More About Automating and Maintaining Impromptu

For more information about automating and maintaining Impromptu, see the *Impromptu Macro Help*.

Chapter 10: Optimizing Performance

This chapter explains how you can optimize Impromptu's performance using

- client/database-server balancing
- table weighting
- qualifications
- governors
- alternative data sources
- user-defined functions
- bulk fetch
- stored procedures
- auditing

Client/Database-Server Balancing Options

With Impromptu's client/database-server balancing features you can optimize processing by determining where and when processing occurs. Setting the correct client/database-server options for different user classes ensures that all users get the best possible performance.

You balance the client/database-server load using settings on the Client/Server tab of the User Profiles dialog box.

You can choose to perform query processing by Database Only, Limited Local Processing, or Flexible Processing. By default, the standard templates provided by Impromptu are set to Limited Local Processing. If your reporting environment requires flexible processing, you can modify the settings in the templates for all new reports that you create.

Database Only

Use the Database Only option to process on the database server only. You might want to use this option when queries are very large, or when you want the Structured Query Language (SQL) generated by queries to be portable to other applications that access your database.

In interactive queries, the Database Only option requires that the query is completely processed on the database server in one query submission. As a result, this option prevents building a query that involves both details and subtotals. This type of option may be suitable for novice users.

This option limits you to executing queries that do not require local processing. All single request queries will be allowed, but not multiple request or multiple-pass queries. For example, a report that includes subtotals and/or grand totals won't be allowed, as these operations are performed in a second request.

This option is useful for SQL requests that will be run in other applications besides Impromptu. You would only see in Impromptu what you were previously able to see in the other application. This option would be a viable solution for users in a warehouse who have diskless Personal Computers (PCs).

If you have the Impromptu Request Server package, then all processing takes place on the database server without restrictions.

Limited Local Processing

Use the Limited Local Processing option when you want data passed from the database server to the PC, which handles processing locally.

In interactive queries, the Limited Local Processing option attempts to perform as much of the processing on the database server as is possible through the use of multiple queries. Impromptu will then combine the results on the PC. Limited Local Processing allows for more advanced type querying than Database Only.

For example, if you want last names changed to uppercase, and you put this condition in the footer of a query, Impromptu applies uppercase to the data as it is displayed. If you put the uppercase condition in the header, Impromptu retrieves all rows in the database and then applies the uppercase condition.

The Limited Local Processing restriction disallows queries that require local sorting. All other queries are allowed. Limited Local Processing won't work if you want to sort on a calculated column requiring local expressions, or if a filter is based on a calculated column. For example, you could not sort or apply a filter on a COUNT, SUM, AVE, MAX or MIN column.

Flexible Processing

With the Flexible Processing option Impromptu determines where processing should take place.

In interactive queries, the Flexible Processing option is the default, and allows Impromptu to determine the location for processing a query. This option allows for the full querying capability of Impromptu. If users require such capability, this is the option you should set.

With this option, depending on the database, the entire result set can be sent to the PC, where local sorts are performed. This option allows Impromptu to execute any query, but it can increase network traffic.

You can also make Client/Database-Server settings at the query level. When you are using a local database, all options on this tab in the Report Query dialog box are dimmed. The dialog displays the message "local database, flexible processing assumed."

Minimize Connect Time to the Database

The Minimize Connect Time to the Database checkbox is below the Query Processing section of the Client/Database tab. If this checkbox is enabled, then Impromptu creates a temporary cache for query results and disconnects the database connection as soon as the report runs. The connection is re-established when another query runs.

Balancing the Client/Database-Server Load: An Example

This example shows how The Great Outdoors handles the problem of Client/Database-Server balancing.

The Problem

Throughout The Great Outdoors user community, reports that you distribute run on diverse environments. Some users have high-performance PCs with lots of processing power and memory. Others have more limited resources. Users also vary in their levels of experience, and in their needs.

The Solution

The Great Outdoors regional and country managers all use high-powered PCs that can handle sophisticated queries and large reports. In addition, regional managers are trained in report creation and database administration, so they are capable of creating their own queries and understanding complex join strategies. In fact, they will want to use the SQL from some of their Impromptu queries in other applications, so they need access to the SQL that Impromptu creates.

Your customers, on the other hand, use low-end 386 machines with little memory. They are interested only in a single report—your product brochure—and they don't want to change this report in any way.

Your sales personnel use high-end notebooks with significant processing power and memory. To meet the needs of these users, you implement the following client/database-server settings for your user classes:

- In spite of their powerful PCs, regional and country managers run reports with database server-only processing. This gives managers access to database server power and generates SQL that is portable to other applications.
- Salespeople use flexible processing, meaning that Impromptu determines when to allow the database server to handle processing and when to perform it on the PC.
- Customers generally have low-end PCs, so you restrict them to limited local processing with most processing done by the database server.

Client/Database-Server Balancing and Summaries

This section describes the Impromptu query model in the context of summaries. Its purpose is to help you understand the differences among the query processing types—Database Only, Limited Local Processing, and Flexible Processing—so you can choose an effective summary/client database-server balancing strategy. It includes examples, and a table to help you determine how the summary mechanism corresponds with Impromptu's Client/Database-Server Balancing options.

Summaries, or aggregates, are calculations that involve a summary component and a FOR clause that indicates the group for which the summary should be calculated. Examples of summaries are: total, minimum, maximum, average, and count.

Introduction

Impromptu adds value on top of the database functions by providing a standard set of query and reporting capabilities, regardless of the underlying database. This allows Impromptu to execute the same queries against all supported data sources, from dBASE, to Informix, to DB2.

These capabilities are modeled internally via a Structured Query Language (SQL) dialect which extends SQL 92, allowing Impromptu to add value to reporting. This SQL dialect is referred to as Cognos SQL.

These Impromptu extensions allow complete reporting flexibility. For example, you could create a report which merged ordered (grouped) details with varying and opposing summary levels. As database SQL does not allow this in a single query, Impromptu provides extended processing in the form of multiple (merged) queries and local processing to derive the required result set. To deliver this flexibility, Impromptu uses a reasonably complex query model with the three client/database-server balancing options described earlier.

The most common requirement is to support both details and summaries in a single report, or to support outer joins beyond those supported by the database. The primary focus on the query model is to provide a tunable mechanism by which to support queries that are too complex to be executed within the limits of the supported database SQL dialect.

Three Options for Mixing Details and Summaries

Impromptu has a data access engine and a report engine. All data access and manipulation is completed before the result set is displayed in a report, so the columns and values displayed in the report must exist in the result set passed from the data access engine to the report engine. You will see how the data access engine and report engine interact to display the result sets in the following Impromptu summary examples. In addition, these examples will help you understand the query processing model.

Option 1 - Running Summaries

One option is to calculate running totals locally as the records are retrieved from the database. For example, as the database records are retrieved to the client, a new column is created where the running total is appended.

In the following example, the user created a report showing Product Type, Product Line, Product, and Product Cost. The user then grouped on Product Type and Product Line, added a footer for Product Line, and placed Total (Product Cost) into the footer, setting the association to Product Line.

Product Type	Product Line	Product	Product Cost
Environmental Line	Alert Devices	Pocket U.V. Alerter	3
		Microwave Detective	4
		Pocket Radon Alerter	13
	Bio-Friendly Soaps	RiverKind Shampoo	3
		RiverKind Soap	4
		RiverKind Detergent	2
	Recycled Products	EnviroSak	2
		Enviro-Kit	4
		Enviro-T	10
	Sunblock	Sun Shelter-8	2
		Sun Shelter-15	3

The query sent to the database is select Product Type, Product Line, Product, Product Cost from ...where ... order by Product Type, Product Line.

The Impromptu report appears as a ‘grouped’ report. However, the grouping is actually SQL ordering combined with the suppressed display of repeating values.

The result set from the database would be:

Product Type	Product Line	Product	Product Cost
Environmental Line	Alert Devices	Pocket U.V. Alerter	3
Environmental Line	Alert Devices	Microwave Detective	4
Environmental Line	Alert Devices	Pocket Radon Alerter	13
Environmental Line	Bio-Friendly Soaps	RiverKind Shampoo	3
Environmental Line	Bio-Friendly Soaps	RiverKind Soap	4
Environmental Line	Bio-Friendly Soaps	RiverKind Detergent	2

In addition to the above database result set, Impromptu creates a running total column (see following table) which is appended to the database result set before being passed from the data access engine to the report engine:

Product Type	Product Line	Product	Product Cost	Total (Product Cost) No. 1
Environmental Line	Alert Devices	Pocket U.V. Alerter	3	3
Environmental Line	Alert Devices	Microwave Detective	4	7
Environmental Line	Alert Devices	Pocket Radon Alerter	13	20

Product Type	Product Line	Product	Product Cost	Total (Product Cost) No. 1
Environmental Line	Bio-Friendly Soaps	RiverKind Shampoo	3	3
Environmental Line	Bio-Friendly Soaps	RiverKind Soap	4	7
Environmental Line	Bio-Friendly Soaps	RiverKind Detergent	2	9

As you can see, the running total column begins re-totalling for each Product Line. Thus, after reading the last record in a group, the Impromptu report engine displays the total.

The generated Cognos SQL creates the database result set using it as a derived table upon which it creates the final result set to pass to the report engine:

```
selectc10 as c1,
    c9 as c2,
    c12 as c3,
    c11 as c4,
    RSUM(c11 for c10,c9) as c5
from
(selectT1."PROD_LINE" as c9,
    T1."PROD_TYPE" as c10,
    T1."PROD_COST" as c11,
    T1."PRODUCT" as c12
from "PRODUCT" T1
order by c10 asc,c9 asc
) D1
```

The italicized portion of the query would be passed to the database after conversion to database native SQL (ODBC in this sample):

```
selectT1.‘PROD_LINE’, T1.‘PROD_TYPE’, T1.‘PROD_COST’,
    T1.‘PRODUCT’
from ‘PRODUCT’ T1
order by 2 asc, 1 asc
```

The beginning portion of the Cognos SQL is the local processing required to create the additional total column in the derived table. The RSUM is the syntax used to create the running summary.

Option 2 - Extended Summaries

A second option is to append the overall group total to the result set using extended summaries. In this case, similar to running summaries, additional columns are appended to the result set. However, these values are calculated for the entire group before being appended to the result set and therefore require more local processing.

Take the same report we looked at a moment ago, and place the total in the header. The total must appear in the first record of each group to ensure the correct value is displayed.

The SQL sent to the database is the same as when using running summaries. However, when the additional column is added, an extended summary is used instead, resulting in:

Product Type	Product Line	Product	Product Cost	Total (Product Cost) No. 1
Environmental Line	Alert Devices	Pocket U.V. Alerter	3	20
Environmental Line	Alert Devices	Microwave Detective	4	20
Environmental Line	Alert Devices	Pocket Radon Alerter	13	20

Product Type	Product Line	Product	Product Cost	Total (Product Cost) No. 1
Environmental Line	Bio-Friendly Soaps	RiverKind Shampoo	3	9
Environmental Line	Bio-Friendly Soaps	RiverKind Soap	4	9
Environmental Line	Bio-Friendly Soaps	RiverKind Detergent	2	9

Notice that now the total must be in the first record of each group so it can be displayed before the details for that group. Note that the same total appears for each record in the group.

The generated Cognos SQL creates the database result set using it as a derived table upon which it creates the final result set to pass to the report engine:

```
selectc10 as c1,
    c9 as c2,
    c12 ac c3,
    c11 as c4,
    XSUM(c11 for c10,c9) as c5
from
(selectT1."PROD_LINE" as c9,
    T1."PROD_TYPE" as c10
    T1."PROD_COST" as c11,
    T1."PRODUCT" as c12
from "PRODUCT" T1
order by C10 asc, C9 asc
) D1
```

Again, the italicized SQL is sent to the database after translation to native syntax:

```
selectT1.‘PROD_LINE’, T1.‘PROD_TYPE’, T1.‘PROD_COST’,
    T1,‘PRODUCT’
from ‘PRODUCT’ T1
order by 2 asc, 1 asc
```

The beginning portion of the Cognos SQL is the local processing required to create the additional total column in the derived table. The XSUM is the syntax used to create the extended summary.

Option 3 - Multiple Queries

A third option is to execute two queries back to the database; one an ordered detail query in the form select, a, b, c, ... from ... where ... order by a, b; and the second a grouped summary query in the form select a, b, sum (C) from ... where ... group by a, b. As both queries are sorted in the same order, Impromptu merges the results locally without re-sorting the two result sets on the PC. This requires very little local processing.

Multiple queries are used only when the report displays the total of a column either before the detail records (i.e., put the total in the group header instead of the group footer), or if the total is displayed in the detail line as a separate column beside the detail column. In other words, this mechanism is not used when the summaries are displayed in a footer after the details.

Recall that the data access and manipulation must be completed before the report is displayed. Thus, if the report displays the total with or before the first detail record, the total must be in the first record of the result set.

```
selectc10 as c1,
    c9 as c2,
    c12 as c3,
    c11 as c4,
    c8 as c5
from
    (select T1."PROD-LINE" as c6,
    T1."PROD_TYPE" as c7,
    SUM(T1."PROD_COST") as c8
from "PRODUCT" T1
group by T1."PROD)TYPE",T1."PROD_LINE"
) D2,
    (select T1."PROD_LINE" as c9,
    T1."PROD_TYPE" as c10,
    T1."PROD_COST" as c11,
    T1."PRODUCT" as c12
from "PRODUCT" T1
) D1
where ((c10 = c7) and (c9 = c6))
order by c1 asc, c2 asc
```

The italicized portion is the two database queries that will be translated and passed to the database.

```
select T1.‘PROD_LINE’, T1.‘PROD_TYPE’, T1.‘PROD_COST’,
T1.‘PRODUCT’
from ‘PRODUCT’ T1
order by 2 asc, 1 asc
select T1.‘PROD_LINE’, T1.‘PROD_TYPE’,
    sum(T1.‘PROD_COST’)
from ‘PRODUCT’ T1
group by T1.‘PROD_TYPE’, T1.‘PROD_LINE’
order by 2 asc, 1 asc
```

The "where" syntax is used to merge (join without sort) the two database result sets locally.

Query Processing Model: Client/Database-Server Balancing

Impromptu uses the Client/Database-Server Balancing options to tune the interaction between the database and the three summary options.

Extended Summaries vs. Multiple Queries

Both extended summaries and multiple query mechanisms are used when the report requires totals to be displayed before the details (in a header) or beside the details (as a column). Impromptu's decision as to which mechanism is used coincides with your choice of client/database-server balancing.

Running Summaries

Running summaries are always used when the summary is displayed after the details.

Summary Mechanisms/Query Processing Options

This is how the summary mechanisms interface with the query processing (Client/Database-Server Balancing) options.

- Database Only. Since the Database Only processing option does not allow mixing details with summaries, this option is not applicable when considering summaries. This option does not allow local processing. Only queries that execute database native SQL are allowed.
- Limited Local Processing. This limits the amount of local processing by disallowing local sorting, and uses multiple queries instead of extended summaries to calculate summaries. In other words, depend on the database to sort and summarize, and use the PC to merge results into a single result set for less costly running totals.
- Flexible Processing. This uses extended summaries instead of multiple queries and allows local sorting.

Summary Table

This table will help you determine how the summary mechanism corresponds with the Client/Database-Server Balancing options.

	Limited Local?	**Flexible Processing?**
Running Summaries: RSUM	YES	YES
Extended Summaries: XSUM	NO	YES
Multiple Database SQL Queries: Sum in second select	YES	NO

Notes

- Since the databases do not allow you to mix details and summaries, the Database Only option is not applicable.
- If the summary is displayed after the details, running summaries are always used.
- If the summary is displayed before or with the details, then either extended summaries or multiple queries are used, depending on the client/database-server balance.

Table Weighting

Table weighting specifies the order in which tables will be retrieved and joined. The more detailed the table, the bigger the weight that should be assigned. The objective is to get Impromptu to read the summary table first, and then join to successively more detailed tables.

By assigning weights to your tables, you control which tables are preferred as intermediate tables in join strategies. Impromptu uses these weights when creating SQL queries by ordering the tables in the FROM clause, from the lightest to the heaviest. Smaller tables are always preferred.

Table Weighting: An Example

A report with Customer (CUSTOMER table), Order_No (ORDER table), and Rep_Name (REP table) selects columns from the CUSTOMER table, then the ORDER table, and then the REP table, because that is the order they are referred to in a query.

By assigning a weight of 20 to the order table, a weight of 10 to the Customer table, and a weight of 2 to the REP table, you determine a more efficient order of the tables in the FROM clause. The Rep table has been given a higher priority (1 being the highest priority). Due to this weighting, tables are selected in the order: REP-ORDER-CUSTOMER.

Why? If direct joins between these tables exist, then data will be selected from the tables in the order based on table weight. However, in this case, there is no direct join between Rep and Customer. They have to be joined through the Order table.

Impromptu creates the table joins in this order:

```
"Rep.Rep_no=Order.Rep_no and Customer.Order_no=Order.Order_no"
```

In many cases, weighting enables you to optimize queries. Advanced databases do a better job of optimizing queries and are unaffected by the order of the tables in the SQL.

Some database optimizers override the weighting option in Impromptu. dBase databases are good candidates for the application of table weighting.

Assigning Weights

For more information about assigning weights, see the Impromptu online Help. From the Help menu, click Contents and Index. Then, in the Find tab, enter "weights" as the keyword for finding information.

Qualification Options

When using more than one database that has a table with the same name, you can use the Qualifications tab of the Tables dialog box to qualify tables in SQL queries. That way users will access the correct tables. By default, tables are fully qualified.

For example, assume that The Great Outdoors company has acquired a small operation that manufactures one of its product lines. When the two sets of data are combined at the database level, the data will contain many of the same names as your data at headquarters. Some of the tables will have the same names but contain different data. You can maintain one catalog for the two regions by qualifying the tables for the different locations, so that the users will access the correct tables even though they are named the same.

Assigning Qualification Settings

For more information about qualifying tables, see the Impromptu online Help. From the Help menu, click Contents and Index. Then, in the Find tab, enter "qualification" as the keyword for finding information.

Governor Options

Using the Governor tab of the User Profiles dialog box, you can make settings that affect processing variables.

Some sorting or indexing activities can degrade performance and produce unpredictable results. As a result, you might want to impose controls on certain user activities, specifying that they are allowed, or that Impromptu warns when they occur, or that Impromptu prevents them from occurring.

These activities include:

- Sorting on Non-Indexed Columns. Impromptu is faster when indexed columns are used to sort data items. For example, if Last_Name is defined as an index in the Employee table, then using it rather than Full_Name would be more efficient.
- Outer Joins. An outer join retrieves all rows in one table even if there is no matching row in another table. For example, if you retrieve all orders by representative, even those representatives that have no orders are retrieved. This type of join can produce very large reports, but some users may need to use this feature. For example, the Sales group would probably want to report all orders by product, including those products that had no sales.

Impromptu provides support for left, right, and full outer joins. In doing so, Impromptu might be required to assist a database which provides less complete support of the SQL-92 definition of outer join processing. This can affect performance. When defining outer joins, please refer to your vendor documentation.

- Suppress Duplicates (Select Distinct). Suppressing duplicate rows in a report can slow processing. This option prevents this. It will affect results on local processing such as RSUM.
- Cross-Product Queries (No Table Joins). Cross-product reports retrieve data from tables without joins. They take a long time to generate, and can produce meaningless results.

Data Access Options

The Governor tab of the User Profiles dialog box also has settings that affect data access options. Because you define them for different user classes, the options can be inherited from higher-level classes by lower-level classes.

These settings include:

- Limit Retrieval of Large Text Items to <n> Characters. This controls the size of BLOBS (Binary Large Objects) that users can retrieve. If your database supports BLOBs, Impromptu truncates the BLOB to the size you specify.
- Limiting the number of records is also required if each record has a large BLOB field. Impromptu lets you restrict data volumes using both the number of records and the size of BLOB fields.

Before the following settings are effective, Impromptu must gather statistics on size and time required to run reports. You might have to run the report a few times to build up the statistical base.

- Report Table Limits. These control the number of tables that a user can retrieve in a report. For new or changed reports, Impromptu counts tables as they are retrieved and issues a warning or stops at the pre-set number. For existing reports, Impromptu stops users from running reports that historically exceed the limits.
- Data Retrieval Limits. These control the number of rows that a user can retrieve. For new or changed reports, Impromptu counts rows as they are retrieved and issues a warning or stops at the pre-set number. For existing reports, Impromptu stops users from running reports that historically exceed the limits.
- Query Execution Time Limits. These limit the time that a query is allowed to take before the user is warned or stopped.

Alternative Data Sources

When Impromptu retrieves information directly from the database, it goes back to the database when you change a query. Selecting an alternative data source can make returning to the database unnecessary, reducing the cost of reporting over your corporate network. Some alternative data options are:

- snapshots
- thumbnails
- HotFiles

Snapshots

A snapshot retrieves all the information for a report and stores it in a permanent cache on the PC. The data is stored along with the report format in the *.imr file.

A snapshot can be used like a database. You don’t need access to the database to use a snapshot. A report can be produced directly from the snapshot. Because you are not accessing the database, you cannot add columns to a snapshot or build a less restrictive filter. Snapshots are suitable for working disconnected from the LAN.

Use snapshots when:

- You want to have all the data you need for your report in one file. When you work with a report that contains a snapshot, you can sort, group, calculate, add filters, and so on.
- You want to e-mail your report as a single file attachment. When you create a snapshot for a report, the snapshot is saved along with the report in the *.imr file. You can send the .imr file to someone who doesn’t have access to your database but who has Impromptu.
- You want to open a report without access to the catalog. Using a snapshot is one way to work with your reports, even when you are not connected to the LAN; for example, when you are using a laptop.

Three Ways to Create a Snapshot

Before you can create a snapshot, you must have a report open. To create a snapshot, you can:

- use the Snapshot button on the Standard PowerBar
- use the Snapshot option in the Access tab of the Query dialog box
- use the Save As command from the File menu

Refreshing a Snapshot

If you had created a snapshot to preserve historical data, you would not update or refresh the snapshot, since it is supposed to be used to preserve the state of the database at a precise moment in time.

However, if you are using a snapshot for other purposes (for example, to work with the data locally, or to e-mail it to someone as a single file attachment), you would want to refresh or update the database to present the new state of the database.

There are two ways to refresh the snapshot:

- use the Snapshot PowerBar button
- use the Access tab of the Query dialog box

You can schedule snapshots using Cognos Scheduler. To do this, run a report and always save it as a snapshot to a different filename using the Report tab in the Scheduler. Saving a report as a snapshot and re-running it results in the data never being updated.

Thumbnails

Impromptu can retrieve a specified number of records from the database and store them in a temporary cache on the PC called a thumbnail.

Impromptu attempts to use the thumbnail exclusively, and does not return to the database unless necessary. When you sort, group, apply restrictive filters, or add columns that are calculations based on existing columns, Impromptu performs these actions locally, without accessing the database.

Unlike snapshots, thumbnails allow you to apply less restrictive filters and add one or more columns, and Impromptu will automatically go back to the original database to get new information. Thumbnails are suitable for testing purposes.

HotFiles

A HotFile is a flat sequential file. It acts as a separate local data table that can be added to your catalog or used in a report as if it were a database table. Both the data and the definitions are stored in the HotFile. They are not indexed, which means that Impromptu must use Sort/Merge and sequential scan to filter.

A HotFile is similar to a snapshot, but unlike a snapshot a HotFile can be:

- added to your catalog and used like a table
- used to create new reports
- used to join multiple databases from the same or different vendors
- used to create multiple query reports
- created automatically using macros

Impromptu's HotFiles enable you to create a generic file that you can use to combine information from multiple databases and database types into a single report. By saving data in a HotFile, you make it available for use with the data in any Impromptu catalog or report.

A HotFile is a separate local data table that can be added to your catalog or used in a report as if it were a regular database table.

You can use a HotFile as a database definition for a catalog. You can also use a HotFile as a table to add to a catalog.

Once you create a HotFile, any report or catalog can use it, even catalogs that use a completely different database than the one from which you created the HotFile.

In addition to giving you simultaneous access to data in multiple heterogeneous databases, HotFiles enable you to:

- Compare historical data. For example, you can compare last month's sales with this month's sales by creating a HotFile for both.
- Access subsets of large tables. For example, if you have millions of rows in a database and you commonly use 20 of them, you can create a HotFile containing only those 20 rows. This saves time and expense when executing queries, because you don't have to scroll through millions of rows.
- Add tables to a database you are using. For example, if your sales commission conversion table is in a separate database, you can add that table to your commissions catalog using a HotFile.
- Speed up processing time by putting calculated and summary data in a HotFile.

Creating a HotFile

When HotFiles are joined to database tables, the query is executed on the database server without the HotFile being considered. The results are brought to the PC and joined to the HotFile.

Use HotFiles when you want to:

- Improve query performance. For example, if you have a large static data set that you use for many reports, you can create a HotFile for this data and use the HotFile data rather than querying the data on the database.
- Improve report execution. If you have a yearly sales table that you use for your financial reports, you can create a HotFile with the subset of data you require and use the HotFile data rather than the database.
- Join multiple databases. If you want to access a sales table from one corporate database, and a product table from a different database, you can create a HotFile of the product table and join the HotFile to the corporate database.
- Create multiple query reports. You can create a complex exception report that requires multiple passes to get the information that you need. You could create a HotFile of your first report (first pass that extracts certain attributes or information), create another Hotfile of any subsequent passes to eliminate or summarize information, and then create the final report from the final HotFile.

Limitations of a HotFile

The issue of joining across database boundaries is not an easy one to address. If you have one database with thousands or millions of rows that you need to combine with another database with thousands or millions of rows, a HotFile is not the answer.

Products such as Sybase OmniServer and Oracle Transparent Gateway allow you to use an existing database server for the joins. Products like Digital's AccessWorks/DBI allow you to use either an existing database server or a separate dedicated database server to join the data. For simpler situations where you have a relatively small or stable set of data that you wish to join to another database, use a HotFile.

For example, if you have Branch information in Oracle (a few hundred records) that you need to join to a large Sybase database, use a HotFile to extract the Branch information and join it to the Sybase database.

For more information about HotFiles, see the Impromptu online Help. From the Help menu, click Contents and Index. Then, in the Contents tab, select

- Create HotFiles

Using HotFiles: An Example

This example shows how The Great Outdoors created a HotFile to overcome the problems associated with using different databases.

The Problem

Each of your regional offices uses its own small database to track regional promotions and sales initiatives. For historical reasons within the company, most of your regions use dBASE for their local database requirements. However, The Great Outdoors corporate information is stored in a Sybase database. When your regional managers receive your standard reports and templates, they need a way to easily incorporate data from the corporate Sybase database with the information in their local dBASE databases.

The Solution

To overcome the multiple database type problems at The Great Outdoors, you create a HotFile called PRODSALE.

This HotFile contains sales information from all regions. You set in place an OLE automation procedure that automatically builds this HotFile at the end of each calendar month, and then mails it to your regional managers. Your regional managers use the information in this HotFile with the information in any of the catalogs they have created for their regional databases. This way, your regional managers can easily correlate their sales results with the costs they have incurred in their regional promotions.

User-Defined Functions

User-Defined Functions (UDFs) are custom functions that you can create and use within SQL statements to query your datasources. You can create UDFs to

- standardize custom functions for your organization
- leverage Impromptu's database support, enabling report users to get data-specific answers from your databases

You can create two kinds of UDFs:

- database UDFs that you can use with one or more of your databases
- external UDFs that are written in C and are compiled in DLLs for use with Windows 95 and Windows NT. For use in UNIX environments (with Impromptu Request Server, for example), external UDFs are compiled in shared libraries. Please refer to your vendor documentation for details on how to create user defined functions in a database to be used in SQL Select statements.

A UDF software development kit (SDK) in available to help you set up functions for use with Impromptu. The kit includes:

- a C source file that contains five user-defined functions
- makefiles for compiling the C source file into a shared library or DLL
- a list of the files that are required to create external UDFs
- a short tutorial on creating UDFs

Before creating your own external functions, be sure to read the online book *How to Create User-Defined Functions*. You can access the book from the Impromptu online Help, or from the Cognos selection in the Windows Start menu.

For more information about user-defined functions, see the Impromptu online Help. From the Help menu, click Contents and Index. Then select "Impromptu User," then "Functions," then "User Defined Functions."

Bulk Fetch

Impromptu attempts to retrieve more than one row of data at a time to reduce processing time. However, if Impromptu attempts to retrieve too many rows at once, your network traffic may increase. If your database supports bulk fetch, Impromptu retrieves 100 rows of data in one fetch call by default.

You can change the maximum number of rows retrieved in each call by typing the following in the Query Options section of the Impromptu.ini file:

```
bulk fetch rows= n
```

See your database documentation for more information about the most efficient bulk fetch settings.

You can monitor performance using the Impromptu auditing tools. For more information about auditing, see "Audit Performance" (p. 88).

Note: You can limit the number of rows each user class can retrieve using the Governor tab (User Profiles dialog box). If the user class limit is less then the bulk fetch limit, Impromptu adheres to the user class limit instead.

Stored Procedures

Impromptu can access stored procedures in your database and show the results in a list report. Stored procedures are SQL scripts stored in your database. They improve the performance of repetitive tasks because they are only compiled the first time you run them.

Impromptu comes with a Stored Procedure template. When you create a new report using the template, the Query dialog box is replaced by the Stored Procedure dialog box. To access the procedure, type the following and click OK:

```
call procedure name (?parameter1? IN, ?parameter2? OUT)
```

You must create a prompt for each input and output parameter your stored procedure requires. For example, a stored procedure requires a customer number and returns the number of tents the customer purchased in March. Use the Prompt Manager to create a prompt for each parameter, and then type the prompts into the call:

```
call sp_custinfo (?custid? IN, ?marchsales? OUT)
```

When you create the prompts, you must give them the same name as the parameter that the stored procedure requires. You must also provide as much information as possible about the type of data the stored procedure returns. You can specify data type information using the Advanced options of the Prompt Definition dialog box.

For more information about specifying the correct datatype, see Chapter 3 of the *Deploying Impromptu Applications* online book.

If you always want to submit the same value for a parameter, you can set the value as a default and use the default every time you run the report. Select the Use the Default Value check box (Prompt Definition dialog box) to use the default value automatically when you run the report.

You can use the results of your stored procedure as a data source for other reports. Save your report, and then create a new report using the same catalog. In the Data Sources box (Query dialog box), you can select Hotfiles and Reports, and browse for the stored procedure report. Or, you can use the Tables dialog box to add the stored procedure report to your catalog.

For more information about stored procedures, see the Impromptu online Help.

Notes

- Stored procedure reports have the same limitations as type-in SQL reports. To remove these limitations, use stored procedure reports as data sources for other reports.

  For more information about type-in SQL, see the Impromptu Administrator online Help.
- If a stored procedure returns more than one result set, Impromptu can only access the first one.
- Impromptu can access stored procedures from any Cognos-supported database. Stored procedures can also be accessed using ODBC connections.

Audit Performance

Impromptu can record information about the tasks you perform in a log file. You can use the information to monitor performance and make decisions about how to improve the efficiency of your Impromptu environment. Impromptu comes with a set of tools that you can use to configure your log file and export the information into a database. Then, you can run a sample Impromptu report on the information to audit Impromptu performance.

For more information about auditing, see the Impromptu Administrator online Help.

For More About Optimizing Performance

For more information about optimizing performance, see the Impromptu online Help. From the Help menu, click Contents and Index. Then select one of

- Create HotFiles
- Manage the Impact on the Network & Database, then "Adjust the Client/Server Balance" or "Assign Weights to Tables"

Chapter 11: Integrating Impromptu with Other Cognos Products

This chapter describes how you can combine Impromptu with Cognos BI, a complete suite of business intelligence tools for your enterprise. Some of these tools, such as Architect and Access Manager, are provided with Impromptu.

You can

- use Architect as an alternative way to create Impromptu catalogs
- prepare reports for distribution over the Web
- drill through to Cognos Query
- import queries from Cognos Query
- drill through from PowerPlay
- create PowerCubes for PowerPlay
- publish reports directly from Impromptu to Upfront, the Cognos Web portal
- use Configuration Manager to control locale settings
- create reports accessible to disabled users

Overview

You can use Cognos BI to give users across your enterprise access to important information quickly, and with easy, centralized administration.

You can use

- Architect for centralized metadata modeling

  Use Architect to create a business model of your database that all your Cognos BI products can use. You can export a part the model as an Impromptu catalog and use it to create reports.
- Impromptu Web Reports to distribute your reports over the Web in a managed environment

  Report consumers can subscribe to reports and use their Web browsers to view them in .pdf, .csv, .html, or .xls format. If they want more information about a certain column in a report, they can drill through to Cognos Query, an ad hoc reporting product for the Web.
- Cognos Query for Web-based reporting

  Cognos Query users can create their own queries. They can sort, filter, group, and format data. Advanced users can create new queries that are then distributed to other users. You can import queries into Impromptu, save them as reports, and distribute them over the Web again using Impromptu Web Reports.
- Access Manager for centralized, enterprise-wide security

  Access Manager uses LDAP technology to provide a single security structure for your entire enterprise.
- PowerPlay for client- and Web-based OLAP reporting

  Use Impromptu to specify a set of data to use as an OLAP data source for PowerPlay. Also, if your users want to see the details behind a trend, they can drill through to an Impromptu report.
- Upfront to distribute reports directly to the Web

  Report consumers can use Upfront NewsBoxes to access reports.
- Configuration Manager to configure Cognos Series 7 components

  Use to manage configuration settings, such as locale and accessibility.

Using Architect to Create a Catalog

You can use Architect to create catalogs for Impromptu. In Architect, you assemble metadata about your databases from your existing sources, and then create a business model that PowerPlay, Impromptu, and Cognos Query can use to access information. You can copy portions of your business model into packages and generate a catalog from each package. You can create several packages from the same business model, each customized for a specific audience. However, because Impromptu connects to one database at a time, you must ensure that all the tables and columns that are referenced in a single package are in the same database.

Catalogs created in Architect use Access Manager to control security. The Architect catalog contains a reference to a namespace on the directory server where Access Manager stores security information. If users aren't listed in that namespace they can't open the catalog. You can use Architect to set up governor options for the Access Manager user classes in the namespace.

For more information about using Architect to create catalogs, see the Architect online Help.

Moving Existing Catalogs Into Architect

You can import your existing catalogs into an Architect model. Architect converts your catalog folders and columns into entities and attributes, and imports condition filters, prompts, and calculations.

Because Architect uses Access Manager for security, ensure the following before importing your catalogs:

- If you use more than one catalog with the same user classes, the user classes must have the same permissions and restrictions.

  For example, the Sales user class in Catalog1.cat must be identical to the Sales user class in Catalog2.cat. If there are discrepancies, you will lose security information when you import the catalogs.
- If different user classes in different catalogs share the same name, rename them before you import the catalogs.

  For example, Catalog1.cat contains a user class called Sales that regional sales administrators use to track the performance of each sales representative. Catalog2.cat contains a user class called Sales that sales representatives use to check their own performance against quota. You should rename the user classes to Sales Admin and Sales Rep.
- The user class names in your catalogs match the user class names in the Access Manager namespace.

  If Architect can't find a user class in the namespace, it discards the security information for that user class when you import the catalog.

For more information about updating to Access Manager, see "Using Access Manager for Security" (p. 61).

Preparing Your Reports for the Web

When your reports are ready for distribution, you can deliver them together with their catalog and any required files to your Impromptu Web Reports report administrator. Your report administrator publishes the reports so that users with the appropriate permission can view them in their Web browsers.

There are several things you should keep in mind when you create reports for distribution over the Web using Impromptu Web Reports. For example, Impromptu Web Reports does not support CognosScript, AutoRun macros, or OLE automation.

For more information about

- posting them on the server
- preparing your reports for the Web

see the Impromptu Administrator online Help.

Drilling Through to Cognos Query

You can use Impromptu to set up drill-through reporting from Impromptu Web Reports to Cognos Query. When you distribute your reports using Impromptu Web Reports, users can drill through on a column that interests them and explore the information on their own.

For example, you distribute a report that lists customer names, order numbers, and the total sale amount for each order. If users want to see transaction details for order number 160, they can click this value. A query, filtered to show only information for order number 160, opens in their Web browser. They can continue to sort, filter, and format the query, and follow hypertext links to related queries.

What Is the Difference Between Impromptu Web Reports and Cognos Query?

Impromptu Web Reports distributes Impromptu reports over the Web. Report consumers can't change the reports. They can select which reports they want to see and schedule reports to run at certain times.

Cognos Query is an ad hoc reporting product for the Web. Users can do many of the same things you can do using Impromptu on your computer, including

- creating their own queries
- sorting, filtering, and formatting queries

Setting Up Drill-Through Access In Cognos Query

When you save a query in Cognos Query, you can create an Impromptu Query Definition file by selecting the Enable Drill Through check box. The .iqd file has the same name as the query, and contains

- the Structured Query Language statement the query is based on
- the name of the database the query attaches to
- a list of the columns in the query and the associated columns in the database

You must select the Enable Drill Through check box to drill through to a query from another Cognos product.

For more information about .iqd files, see the Cognos Query documentation.

Setting Up Drill-Through Access in Impromptu

In Impromptu, you can select which Cognos Query server you want to drill through to using the Drill Through tab (Options dialog box). Then associate a column in your report with an .iqd file on that server using the Drill Through Properties dialog box.

The Impromptu report and the query in Cognos Query don't have to be related. You can associate a column in your report with any .iqd file on the Cognos Query server and drill through to it. However, your Impromptu catalog and the Cognos Query model should attach to the same database and have at least one common column for drilling through to be meaningful. To ensure your applications use the same data source, use Architect as your common modeling tool.

Before you distribute the report in Impromptu Web Reports, you can test the drill-through access in Impromptu. When you click the Drill Through button, Impromptu creates the LaunchCQ.html file in your temporary folder and opens the file in your default Web browser. LaunchCQ.html contains

- scripts that launch Cognos Query
- a form that automatically runs the query contained in the .iqd file
- filter statements so that the query shows only information about the drill-through column

When you distribute your report over the Web, Impromptu Web Reports shows drill-through columns as hypertext links.

For more information about drilling through to Cognos Query, see the Impromptu online Help.

Importing Queries From Cognos Query

If you want to distribute a query from Cognos Query in a managed reporting environment, you can open it as a report in Impromptu and distribute it using Impromptu Web Reports.

In Cognos Query, you export the query as an Impromptu report. Cognos Query creates a .cq file that contains

- the name of the package in Architect that the query is based on
- script language that tells Impromptu how to recreate the query

Before you use the .cq file to create a report in Impromptu, you must create a catalog for the new report to use. In Architect, export the package that the query is based on as an Impromptu catalog.

In Impromptu, you use the Import Cognos Query dialog box (Tools menu) to select the .cq file and the catalog. Impromptu reads the script, runs the query, and shows the result in a list report. You can change or format the query, and then redistribute it using Impromptu Web Reports.

For more information about importing queries from Cognos Query, see the Impromptu online Help.

Drilling Through from PowerPlay

PowerPlay users can select a value in a PowerPlay report and obtain more information about this value by drilling through to Impromptu. For example, a PowerPlay report shows that Environmental Line sales in January 1998 are $40,000. PowerPlay users can drill through to Impromptu and get the order details for this $40,000 in sales.

How Does PowerPlay Drill Through Work?

PowerPlay users drill through to a saved Impromptu report. The Impromptu report is associated with a PowerPlay measure such as sales, cost, or profit margin percentage in PowerPlay Transformer.

When a PowerPlay user chooses the Drill Through command, Impromptu starts, the associated report opens, and information passes from PowerPlay to Impromptu.

Drill-through information is applied to the associated report when applicable. For example, a PowerPlay report shows sales for the Environmental Line in January 1997. The information passed from PowerPlay to Impromptu is Product = "Environmental Line" and Year = "Jan97." The report associated with the sales measure appears with the filter "Product = ‘Environmental Line’ and Year = ‘Jan97’" applied to the report data in addition to any pre-existing filters.

If Product and Year are not data items in the associated report, the filter cannot be applied and all the information in the report appears. If Product is a data item in the associated report and Year is not, Year = "Jan97" is dropped from the filter.

Impromptu matches the filter information it receives from PowerPlay to columns in the Impromptu Data tab (Query dialog box). If the Year column has been changed in the Query Data tab to "MthYear", Year = "Jan97" is dropped from the filter (even when Year appears as a column title in the associated report).

The filter information passed from PowerPlay is not visible in the Impromptu expression editor. However, it can be viewed in the Impromptu-generated SQL.

Notes

- The Impromptu report cannot be saved as a snapshot by the PowerPlay user.
- The Impromptu drill-through report cannot be a crosstab report.

How Are Measures and Impromptu Reports Associated?

To enable drill-through access in PowerPlay reports, a PowerPlay measure is associated with an Impromptu report when the PowerCube is created.

While all measures in a PowerCube can be associated with a single Impromptu report, each measure is typically associated with a different Impromptu report. For example, the Revenue measure can be associated with a sales report, and the Inventory measure can be associated with a stock report.

The association of measures and Impromptu reports is flexible. For example, more than one Impromptu report can be associated with a measure, and not all measures in a PowerCube are associated to an Impromptu report.

If there is more than one report associated with a measure, a list of reports and their corresponding descriptions appears when a PowerPlay user initially chooses the Drill Through command. When the user selects a report, it becomes the default report for the current measure.

Setting Up Drill Through Access

To enable drill-through access from PowerPlay to Impromptu, an administrator saves an Impromptu report as

- an Impromptu Query Definition (.iqd) file

  PowerPlay Transformer uses the .iqd file as a data source for building PowerCubes.
- an Impromptu Report (.imr) file

  The Transformer administrator associates a measure in the PowerCube with the report.

Notes

- More than one .iqd file can contribute to a PowerCube if you have PowerPlay 5.0 or later installed.
- Binary Large Objects (BLOBS) are not supported in PowerPlay Transformer, and they cannot be part of the .iqd or .imr files for PowerPlay.
- Character or numeric values converted to date values for a PowerCube may cause Impromptu to generate invalid SQL for these columns when PowerPlay users drill through to Impromptu. To prevent this, ensure that date values for Transformer originate from date values in the underlying data source.

For more information about drilling through to Impromptu from PowerPlay, see the PowerPlay online Help.

Creating PowerCubes

Your PowerPlay administrator can use your Impromptu reports as data sources for PowerCubes. PowerCubes are OLAP data sources that you can open in PowerPlay to explore multidimensional information.

To use your report as a PowerPlay data source, you must save it as an .iqd file. The .iqd file contains information about how to connect to the database and which columns are in the report. Your PowerPlay administrator opens the .iqd file in PowerPlay Transformer and models the data into a PowerCube.

When you create a report as a data source for Transformer, keep the following points in mind:

- Create simple reports.
- Do not use crosstab reports.
- Include only those columns in the report that you want PowerPlay to use.
- Use filters to exclude information you do not want to view in PowerPlay.

Creating PowerCubes Automatically

You can use the Create PowerPlay PowerCube button to automatically create a PowerCube based on the currently open report. When you click the button, Impromptu runs a macro that saves your report as an .iqd file, starts Transformer, generates the PowerCube, and opens the PowerCube in PowerPlay.

Transformer uses the AutoDesign command to generate the PowerCube. AutoDesign decides which columns to use as dimensions and which to use as measures. The following are tips you can follow to ensure AutoDesign works well:

- Use real date fields, not numeric or character fields containing date values.

  AutoDesign requires real date fields in order to break dates down by calendar year, quarter, and month.
- Give related columns similar names.

  AutoDesign assumes that columns with similar names are more likely to be related.
- Scan more data for a more accurate PowerCube.

  The more data PowerPlay scans, the more accurate the PowerCube data. However, creation time is longer for larger scan volumes. To change the number of records scanned, edit the following entries in the Trnsfrmr.ini file:

  rowCheckMax=600

  sampleMax=300

  The maximum sampleMax value is 1000.

Publish Reports to Upfront

You can publish reports to Upfront, the Cognos Web portal, to make them available to users as NewsItems in HTML, PDF, Excel and CSV format. NewsItems are organized in NewsBoxes.

You can publish reports directly to Upfront, using the Impromptu Web Reports server. The computer on which Impromptu is installed must be configured to use the same security, or directory server, as the computer on which Impromptu Web Reports is installed.

A report must be saved before you can publish it to Upfront.

Steps

1. With a report open, from the File menu, click Publish to Upfront.

   The Publish to Upfront: Connect to IWR Server dialog box appears.
2. In the URL of IWR Gateway box, enter the URL location for the IWR server. The URL has the form http://*serve*rname/*web_alias*/cgi-bin/*gateway_connector.*

   There are 3 gateway connectors, one for each type of web server protocol.

Web server protocol	Gateway connector
CGI	imrap.cgi
ISAPI	imrapisapi.dll
NSAPI	imrapnsapi.dll (Windows) imrapnsapi.so (Solaris, AIX) imrapnsapi.sl (HP)

3. Click OK.

   The Publish Report Projects1 dialog box appears.
4. Under Publish, choose whether to publish the current report or the entire report set.
   - To publish only the current report, click Only the Current Report.
   - To publish all the Impromptu application files found in the folder specified in the Publish box, click All Reports At.

5. In the Report Set Options box, specify the report parameters for publishing.

Select	To
Update the Existing Report Set	Update a previously published report or report set
Create a New Report Set Named	Specify the folder location for a new report or report set. Impromptu creates the report files for publishing in this location. Enter the source location of the reports to publish using the IWR Server Report Set Location box.

6. Under Publish Reports to Upfront NewsBox, click Select.
   The Select Newsbox dialog box shows the NewsBoxes that exist in Upfront.
7. Click the Upfront NewsBox in which you want to publish the report or report set.
8. Click OK.
   The NewsBox location appears in the Publish Reports to Upfront NewsBox box.
9. In the Publish Report Project1 dialog box, click OK.
   You are notified when publishing is complete.
   To view the results, open Upfront and locate the reports in the NewsBox you specified in Impromptu.

Note

- Report sets are updated in the same manner as report sets are updated by IWR Report Administrator.

Locale Settings

Locale settings determine how locale-specific data are formatted, such as numbers, dates, and currency. For more information about how locale is determined, see the *Planning Advanced Installations Guide*. For more information about modifying locale settings, see the *Configuration Manager User Guide*.

Creating Accessible Reports

Cognos is committed to assisting people with disabilities, and promotes initiatives that make workplaces and technologies accessible. With Series 7 Version 2, Cognos introduces its report-reading solution for accessibility.

Enabling Accessibility Support

By default, reports created in Impromptu are not accessible to disabled users. To enable accessibility, you must specify the appropriate settings in Configuration Manager.

Reports rendered with accessible tags can be read by screen readers when the reports exist in PDF format.

Steps

1. Open Configuration Manager.
2. From the File menu, click Open the current configuration.
   The current configuration of your computer appears in the Configuration Manager window.
3. From the View menu, click Advanced.

4. In the Explorer, under Cognos Impromptu, set accessibility for the appropriate component.
   - To set accessibility for Impromptu User, under Impromptu, click the Accessibility category.
   - To set accessibility for Impromptu Administrator, under Impromptu Administrator, click the Accessibility category.
5. Click the Accessible PDF property, and change the Value property to True.
6. From the Actions menu, click Validate Selection.
7. Click the Accessibility category, then from the Actions menu, click Apply Selection.

Note

- PDF documents that support software accessibility guidelines contain additional markup information that increases the size of the PDF file produced and, as a result, may impact performance. For more information about performance and accessibility, see the *Planning Advanced Installations Guide*.

Tip

- If a report with accessible information degrades performance, render the same report twice. Render one report with accessible information and the other without accessible information.

Design Considerations for Accessible Reports

When creating accessible reports for people with disabilities, consider the following design "do's and don'ts":

- Avoid using visual cues such as text bolding or color to convey important information.
- Avoid using pictures and OLE objects in PDF documents.

  These objects are tagged as artifacts and ignored by screen readers.
- Avoid using conditional formatting to convey important information.
- If using a chart, ensure that a table is included, and that the table provides the same information as the chart.

In addition to these guidelines, you should be aware of report authoring features that are not supported for accessible reports in your product.

The following Impromptu report authoring features are not supported for accessible reports:

- exception highlighting
- verbalization of images
- verbalization of the functionality of horizontal or vertical scroll bars

In addition, accessibility to complex crosstab displays is limited, due to current limitations with assistive technology.

Index